

Revolution Reference

Version 2.1.2

\$

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

@ keyword, function control structure, getProp control structure, on control structure, param function, paramCount function, params function, setProp control structure, How to get the location of the user's home directory

Summary

The character \$ (dollar sign) is used to indicate an environment variable on Unix systems and a command-line parameter on Unix or Windows systems.

Syntax

Example Code

```
put $LOGNAME into field "Login Name"  
if $0 is not myAppName then answer "Problem initializing!"
```

Comments

Use the \$ keyword to interact with the system environment and to find out what arguments were used if the application was started up from the command line.

Comments:

The \$ character marks two kinds of special variables: command-line arguments (on OS X, Unix, and Windows systems) and environment variables (on OS X and Unix systems).

If you start up the application from the command line (on OS X, Unix or Windows systems), the command name is stored in the global variable \$0 and any arguments passed on the command line are stored in numbered variables starting with the \$ character. For example, if you start the application by typing the following shell command:

```
myrevapp -h name
```

then the global variable \$0 contains "myrevapp" (the name of the application), \$1 contains "-h", and \$2 contains "name".

If an argument includes spaces, it must be enclosed in quotes on the command line:

```
myrevapp -in "new info.txt" -out "new info.xml"
```

Gotcha: On Windows XP systems, individual arguments passed on the command line are placed in separate variables (\$1, \$2, and so on) only if they are quoted on the command line. Otherwise, all arguments are placed in the \$1 variable.

On Unix and OS X systems, a variable whose name begins with the \$ character is exported to the application's environment, and is inherited by processes started up by the shell function or the open process command. Use this technique to create your own environment variables.

You can access existing environment variables by prepending the \$ character to the environment variable's name. For example, the following statement gets the contents of the LOGNAME environment variable:

```
get $LOGNAME
```

&

operator

Synonyms

Objects

string

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

&& operator, , operator, combine command, Operator Precedence Reference, Recipe for adding a prefix to each line of a string, Recipe for an Elizabethan insult generator, Recipe for building a repeated string

Summary

Concatenates two strings.

Syntax

string1 & string2

Example Code

```
put "foo" & "bar" -- evaluates to "foobar"  
put myVar & return & return into otherVar  
get offset(return & space, theData)
```

Comments

Use the & operator to create a single string out of two or more parts.

Parameters:

The operands string1 and string2 are literal strings of characters (delimited with double quotes), or expressions that evaluate to strings.

Comments:

The result of the & operator is a string consisting of string1 followed by string2.

&&

operator

Synonyms

Objects

string

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

& operator, , operator, Operator Precedence Reference, Recipe for an Elizabethan insult generator, Recipe for finding lines that are not common to two containers

Summary

Concatenates two strings and inserts a space between them.

Syntax

string1 && string2

Example Code

```
put "my" && "house" -- evaluates to "my house"
put field "First" && field "Last" into field "Name"
find it && "Card"
```

Comments

Use the && operator to combine two strings with a space between them—for example, to combine two words or phrases.

Parameters:

The operands string1 and string2 are literal strings of characters (delimited with double quotes), or expressions that evaluate to strings.

Comments:

The result of the && operator is a string consisting of string1, a space, and string2.

The && operator is equivalent to & space &. In other words,

"this" && "that"

does the same thing as

"this" & space & "that"

()

operator

Synonyms

Objects

logical, numeric, string

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

* operator, + operator, - operator, [] keyword, / operator, value function, Operator Precedence Reference

Summary

Groups operands together.

Syntax

(expression)

Example Code

```
get (quantity * priceEach) + shippingCost  
("a" is within field 1) or ("b" is within field 2)  
23 * ((4 / 17) + 60) + (- 7)
```

Comments

Use parentheses () to group operands together in order to control the order operations are performed in, or to make the structure of a complex expression clearer.

Parameters:

The expression is any Transcript expression.

Comments:

When Revolution evaluates an expression, operations enclosed within parentheses are performed first. If parentheses are nested, the expression within the innermost set of parentheses is evaluated first.

For example, the sin function is evaluated before the / operator, so the sin of 1/4 means "take the sine of one, then divide by four". To obtain the sine of 1/4, use parentheses to force the division to be done first, as in this expression: the sin of (1/4).

Even when they are not needed to change operator precedence, parentheses are useful in making complex expressions more readable. For example,
(quantity * priceEach) + (shippingCost * weight)

evaluates to the same number as

`quantity * priceEach + shippingCost * weight`

But the first example is easier to understand, because the parentheses break down the expression logically for the reader.

Parentheses in expressions must be used in pairs, each (with a matching). Use of unmatched parentheses will cause a compile error.

operator

Synonyms

Objects

numeric

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

/ operator, ^ operator, multiply command, Operator Precedence Reference

Summary

Multiplies two numbers.

Syntax

number1 * number2

Example Code

```
get 3 * 5 -- evaluates to 15
put thisNumber * it into field "Result"
put (3 + commonFactor) * 4 into tempVariable
```

Comments

Use the * (times) operator to multiply two numbers.

Parameters:

The operands number1 and number2 are numbers or expressions that evaluate to numbers, or arrays containing numbers.

Comments:

To multiply the contents of a container by a number, use the multiply command instead.

If either number1 or number2 is an array, each of the array elements must be a number. If an array is multiplied by a number, each element is multiplied by the number. If an array is multiplied by an array, both arrays must have the same number of elements and the same dimension, and each element in one array is multiplied by the corresponding element of the other array.

If an element of one array is empty, the * operator treats its contents as zero.

Changes to Transcript:

The option to multiply arrays was introduced in version 1.1. In previous versions, only single numbers could be used with the `*` operator.

+

operator

Synonyms

Objects

numeric

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

- operator, add command, Operator Precedence Reference

Summary

Adds two numbers.

Syntax

number1 + number2

Example Code

```
put 2 + 2 into fourContainer  
set the layer of field myNumber to (myNumber + 2)
```

Comments

Use the + (plus) operator to add two numbers, or to add two arrays containing numbers.

Parameters:

The operands number1 and number2 are literal numbers, or expressions that evaluate to numbers, or arrays containing numbers.

Comments:

To add a number to the contents of a container, use the add command instead.

If either number1 or number2 is an array, each of the array elements must be a number. If a number is added to an array, the number is added to each element. If an array is added to an array, both arrays must have the same number of elements and the same dimension, and each element in one array is added to the corresponding element of the other array.

If an element of one array is empty, the + operator treats its contents as zero.

Changes to Transcript:

The option to add arrays was introduced in version 1.1. In previous versions, only single numbers could be used with the + operator.

,
operator

Synonyms

Objects

string

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

& operator, && operator, Operator Precedence Reference

Summary

Concatenates (joins) two strings and inserts a comma between them.

Syntax

string1 , string2

Example Code

```
put "first","second" -- evaluates to "first,second"  
get lastName,firstName && middleName
```

Comments

Use the , operator to combine two strings with a comma between them.

Parameters:

The operands string1 and string2 are literal strings of characters (delimited with double quotes), or expressions that evaluate to strings.

Comments:

The result of the , operator is a string consisting of string1, a comma, and string2.

The , operator is equivalent to & comma &. In other words,

"this" , "that"

does the same thing as

"this" & comma & "that"

-
operator

Synonyms

Objects

numeric

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

+ operator, -- keyword, subtract command, Operator Precedence Reference

Summary

Subtracts one number from another, or designates a number as negative.

Syntax

firstNumber - secondNumber

numberArray - {number | array}

-negativeNumber

Example Code

```
put -24 into myContainer  
put theRight - theLeft into theWidth  
put myArray - 17 into adjustedValues
```

Comments

The - (minus) operator serves two purposes. When it has a single operand (unary minus), it negates that number. When it has two operands (binary minus), it subtracts the second number from the first number.

Parameters:

The operands firstNumber and secondNumber are numbers, or expressions that evaluate to numbers, or arrays containing numbers.

Comments:

To subtract a number from the contents of a container, use the subtract command.

You cannot use the unary minus twice in a row. The expression

- - someNumber

causes an error, and the expression

-- someNumber

is interpreted as a comment because it starts with the comment keyword --.

If firstNumber or secondNumber is an array, each of the array elements must be a number. If a number is subtracted from an array, the number is subtracted from each element. If an array is subtracted from an array, both arrays must have the same number of elements and the same dimension, and each element in one array is subtracted from the corresponding element of the other array.

If an element of an array is empty, the - operator treats its contents as zero.

The unary minus cannot be used with an array.

Changes to Transcript:

The option to subtract arrays was introduced in version 1.1. In previous versions, only single numbers could be used with the - operator.

--

keyword

Synonyms

#

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

- operator, `/**/` keyword, How to include comments and notes in a script, How to temporarily remove a portion of a script, Script menu > Comment (Script Editor), Script menu > Uncomment (Script Editor), View menu > Default Comment Character... (Script Editor)

Summary

Indicates the start of a comment.

Syntax

Example Code

```
put 2 into myVar -- myVar holds the index
-- everything on this line is a comment
```

Comments

Anything between -- and the end of the current script line is treated as a comment and is ignored by Revolution when executing the handler.

Comments:

Comments are useful for documenting and explaining your code, either for others who might need to read and modify it, or for yourself. (The code may be clear in your mind now, but in six months, you'll be glad you included comments.)

Comments can be placed anywhere in a script—inside handlers or outside all handlers. In a long script with many handlers, it may be useful to divide the handlers into sections. Each section starts with a comment containing the section name and any other useful information. This practice helps you keep long scripts organized. Similarly, a lengthy handler can be made more readable by explanatory comments.

Comments can contain any text, including lines of Transcript code. If the code is within a comment, it's ignored. You can temporarily remove sections of code for debugging by putting those sections inside a comment.

/

operator

Synonyms

Objects

numeric

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

* operator, div operator, divide command, mod operator, Operator Precedence Reference

Summary

Divides one number by another number.

Syntax

number/divisor

Example Code

```
put 22/7 into approxPi -- approximates pi
get thisVariable/(thisVariable + 1)
```

Comments

Use the / (divide) operator to divide one number by another.

Parameters:

The operands number and divisor are numbers, or expressions that evaluate to numbers, or arrays containing numbers.

Comments:

To divide the contents of a container by a number, use the divide command instead.

If number is an array, each of the array elements must be a number. If an array is divided by a number, each element is divided by the number. If an array is divided by an array, both arrays must have the same number of elements and the same dimension, and each element in one array is divided by the corresponding element of the other array.

If an element of an array is empty, the / operator treats its contents as zero.

Attempting to divide by zero causes an execution error.

Changes to Transcript:

The option to divide arrays was introduced in version 1.1. In previous versions, only single numbers could be used with the / operator.

//**

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

-- keyword, How to include comments and notes in a script, How to temporarily remove a portion of a script

Summary

Delimits a block comment.

Syntax

Example Code

```
answer whichPrompt /* use the prompt that was set earlier */  
/* This entire block, although it is two lines  
   long, is a single comment. */
```

Comments

Anything between `/*` and `*/` is treated as a comment and is ignored by Revolution when executing the handler.

Comments:

Comments are useful for documenting and explaining your code, either for others who might need to read and modify it, or for yourself. (The code may be clear in your mind now, but in six months, you'll be glad you included comments.)

The block comments created by `/**/` differ from the line comments created by `--` because a block comment can span multiple lines, as well as a single line or part of a line. (A comment marked by `--` extends only to the end of the line.) A comment that starts with `/*` does not end until `*/`, even if there are several lines in between.

Comments can be placed anywhere in a script—inside handlers or outside all handlers. In a long script with many handlers, it may be useful to divide the handlers into sections. Each section starts with a comment containing the section name and any other useful information. This practice helps you keep

long scripts organized. Similarly, a lengthy handler can be made more readable by explanatory comments.

Comments can contain any text, including lines of Transcript code. If the code is within a comment, it's ignored. You can temporarily remove sections of code for debugging by putting those sections inside a comment.

;
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

keyword, scriptLimits function

Summary

The character ; is used to place two statements on a single line.

Syntax

Example Code

```
go next card; wait 1 second; go previous card  
repeat with x = 1 to 10; doSomething x; end repeat
```

Comments

Use the ; character to compress code into fewer visible lines for easier reading.

Comments:

Lines that are split with ; are shown in the script editor as a single line, but when executed, are treated as multiple lines of code. The following line counts as three statements:

```
go card 1; beep 2; answer the date
```

A ; character which is used within a literal string does not signal a new line, because the ; is treated as part of the string instead of being treated as a line break.

<

operator

Synonyms

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

<= operator, > operator, caseSensitive property, max function, min function, Operator Precedence Reference

Summary

Compares two values and returns true if the first value is less than the second value, false otherwise.

Syntax

value1 < value2

Example Code

```
3 < 4 -- evaluates to true
7 < (2 + 1) -- evaluates to false
if thisVariable < 0 then beep
```

Comments

Use the < (less than) operator to compare two numbers or to compare the alphabetical order of two strings.

Parameters:

The operands value1 and value2 can be numbers, literal strings of characters (delimited with double quotes), or any sources of value.

Comments:

When comparing strings, the < operator compares the two values character by character, using the ASCII value of each character. For example, "a" comes before "b" in the ASCII character set, so "a" < "b" and "ab" < "bb".

If the strings are of different lengths, so that the trailing characters in one string are compared to missing characters in the other, the missing characters are considered to have lower value than any character. For example, "ab" < "abc".

If the two values are equal, firstValue < secondValue evaluates to false.

If the `caseSensitive` property is true, the comparison between two strings treats uppercase letters as coming before lowercase letters, so `"A" < "a"`. If the `caseSensitive` property is false, the comparison is not case-sensitive, so `"a"` is considered equivalent to `"A"`.

<=

operator

Synonyms

£

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

< operator, >= operator, caseSensitive property, max function, min function, Operator Precedence Reference, Why does a handler stop working when moved to another platform?

Summary

Compares two values and returns true if the first value is less than or equal to the second value, false otherwise.

Syntax

value1 <= value2

Example Code

```
22 <= 21 -- evaluates to false
3 <= -3 -- evaluates to false
"a" <= "a" -- evaluates to true
```

Comments

Use the <= (less than or equal to) operator to compare two numbers or to compare the alphabetical order of two strings.

Parameters:

The operands value1 and value2 can be numbers, literal strings of characters (delimited with double quotes), or any sources of value.

Comments:

When comparing strings, the <= operator compares the two values character by character, using the ASCII value of each character. For example, "a" comes before "b" in the ASCII character set, so the following are all true:

```
"a" <= "a"
"a" <= "b"
"ab" <= "bb"
```

If the strings are of different lengths, so that the trailing characters in one string are compared to missing characters in the other, the missing characters are considered to have lower value than any character. For example, "abc" <= "ab" is false.

If the caseSensitive property is true, the comparison between two strings treats uppercase letters as coming before lowercase letters. If the caseSensitive property is false, the comparison is not case-sensitive, so "a" is considered equivalent to "A".

Cross-platform caution! The synonym £ can be used only on Mac OS and OS X systems. If you use a script containing the £ character on a Windows or Unix system, a script error may result. To ensure cross-platform compatibility, use the synonym <= instead.



operator

Synonyms

`_`, is not

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

< operator, = operator, > operator, caseSensitive property, contains operator, Operator Precedence Reference, Why does a handler stop working when moved to another platform?

Summary

Compares two values and returns true if they are not equal, false if they are equal.

Syntax

value1 <> value2

Example Code

```
3 + 2 <> 6 -- evaluates to true
"abc" <> "abd" -- evaluates to true
field "Old Password" <> field "Password"
```

Comments

Use the <> (inequality) operator to compare two numbers or to compare two strings.

Parameters:

The operands value1 and value2 can be numbers, literal strings of characters (delimited with double quotes), or any sources of value.

Comments:

When comparing strings, the <> operator compares the two values character by character, using the ASCII value of each character. If the caseSensitive property is true, the comparison between two strings treats uppercase letters as coming before lowercase letters, so "A" <> "a". If the caseSensitive property is false, the comparison is not case-sensitive, so "a" is considered equivalent to "A".

Cross-platform caution! The synonym `_` can be used only on Mac OS and OS X systems. If you use a script containing the `_` character on a Windows or Unix system, a script error may result. To ensure cross-platform compatibility, use one of the synonyms <> or is not instead.

=

operator

Synonyms

is

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

<> operator, caseSensitive property, contains operator, Operator Precedence Reference

Summary

Compares two values and evaluates to true if they are equal, false if they are not equal.

Syntax

value1 = value2

Example Code

```
1 = 0 -- evaluates to false
```

```
17 + 9 = 26 -- evaluates to true
```

```
"ABC" = "abc" -- true if and only if caseSensitive is false
```

Comments

Use the = (equality) operator to find out whether two numeric expressions yield the same number or whether two strings are equivalent.

Parameters:

The operands value1 and value2 can be numbers, literal strings of characters (delimited with double quotes), or any sources of value.

Comments:

When comparing strings, the = operator compares the two values character by character. If the caseSensitive property is true, the comparison between two strings treats uppercase letters as coming before lowercase letters. If the caseSensitive property is false, the comparison is not case-sensitive, so "a" = "A".

>

operator

Synonyms

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

< operator, >= operator, caseSensitive property, max function, min function, Operator Precedence Reference

Summary

Compares two values and returns true if the first value is greater than the second value, false otherwise.

Syntax

value1 > value2

Example Code

```
1 > 0 -- evaluates to true
2 > -15 -- evaluates to true
repeat while counter > 0
```

Comments

Use the > (greater than) operator to compare two numbers or to compare the alphabetical order of two strings.

Parameters:

The operands value1 and value2 can be numbers, literal strings of characters (delimited with double quotes), or any sources of value.

Comments:

When comparing strings, the > operator compares the two values character by character, using the ASCII value of each character. For example, "a" comes before "z" in the ASCII character set, so "z" > "a" and "az" > "ab".

If the strings are of different lengths, so that the trailing characters in one string are compared to missing characters in the other, the missing characters are considered to have lower value than any character. For example, "DEF" > "DE".

If the two values are equal, firstValue > secondValue evaluates to false.

If the `caseSensitive` property is true, the comparison between two strings treats uppercase letters as coming before lowercase letters, so `"a" > "A"`. If the `caseSensitive` property is false, the comparison is not case-sensitive, so `"a"` is considered equivalent to `"A"`.

`>=`

operator

Synonyms

—

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

`<=` operator, `>` operator, max function, min function, Operator Precedence Reference, Why does a handler stop working when moved to another platform?

Summary

Compares two values and returns true if the first value is greater than or equal to the second value, false otherwise.

Syntax

`value1 >= value2`

Example Code

```
22 >= 23 -- evaluates to false
```

```
myValue >= 0 -- evaluates to whether myValue is non-negative
```

Comments

Use the `>=` (greater than or equal to) operator to compare two numbers or to compare the alphabetical order of two strings.

Parameters:

The operands `value1` and `value2` can be numbers, literal strings of characters (delimited with double quotes), or any sources of value.

Comments:

When comparing strings, the `>=` operator compares the two values character by character, using the ASCII value of each character. For example, "z" comes after "a" in the ASCII character set, so the following are all true:

```
"z" >= "z"
```

```
"z" >= "a"
```

```
"zz" >= "za"
```

If the strings are of different lengths, so that the trailing characters in one string are compared to missing characters in the other, the missing characters are considered to have lower value than any character. For example, "abc" >= "ab".

If the caseSensitive property is true, the comparison between two strings treats uppercase letters as coming before lowercase letters. If the caseSensitive property is false, the comparison is not case-sensitive, so "a" is considered equivalent to "A".

Cross-platform caution! The synonym _ can be used only on Mac OS and OS X systems. If you use a script containing the _ character on a Windows or Unix system, a script error may result. To ensure cross-platform compatibility, use the synonym >= instead.

@

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

\$ keyword, function control structure, getProp control structure, on control structure, param function, paramCount function, params function, setProp control structure

Summary

The character @ (at sign) is used with a parameter declaration, to indicate that a reference to the parameter is passed instead of its value.

Syntax

Example Code

```
on myHandler thisParameter,@thatParameter
```

Comments

Pass a parameter by reference when you want a handler to change a variable in the calling handler, or when you want a handler to return more than one value.

Comments:

Parameters to a handler are declared on the first line of the handler. If the name of a parameter is preceded with the @ character, that parameter's value is interpreted as a variable name, rather than the value in the variable. Changing the parameter variable in the called handler changes the value of the variable in the calling handler.

The following handler takes a parameter and simply adds 1 to it:

```
on setVariable @incomingVar -- notice the @ before the parameter name
  add 1 to incomingVar
end setVariable
```

The following handler calls the "setVariable" handler above:

```
on mouseUp
```

```
put 8 into someVariable  
setVariable someVariable  
answer "someVariable is now:" && someVariable  
end mouseUp
```

Because the parameter for the "setVariable" handler is declared with a leading @, the mouseUp handler passes "someVariable" by reference. This means that when the "setVariable" handler makes changes to the parameter, it changes the actual variable, and those changes affect all further references in the mouseUp handler to the variable. Executing this mouseUp handler displays a dialog box that says "someVariable is now: 9".

[]

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

() operator, combine command, intersect command, keys function, split command, union command, About containers, variables, and sources of value, How to access a value in an array variable, How to change an element in an array variable, How to create an array variable, How to find out the number of elements in an array, How to find out whether a container is an array

Summary

The square bracket characters [] surround the element name in an array reference.

Syntax

Example Code

```
put thisValue into myArray["someKey"]
if myArray["someOtherKey"] is "A" then beep
```

Comments

Use square brackets [] to specify which element in an array you are referring to.

Comments:

You access an element of an array by using the name of the array, along with the key in square brackets:

```
put "A" into myList["firstLetter"]
put "B" into myList["secondLetter"]
put "C" into myList["thirdLetter"]
get myList["secondLetter"] -- yields "B"
```

Square brackets must be used in pairs, each [with a matching]. Using a single square bracket does not cause an error, but the bracket is interpreted as part of the variable name, rather than as the marker for the key of an array reference.

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

& operator, && operator, ; keyword, Why does a handler stop working when moved to another platform?

Summary

The character is used to break a line in a script for display, while still having it treated as a single statement.

Syntax

Example Code

```
answer "You've been waiting for" && numberOfMinutes  
      && "minutes." with "Keep Waiting" or "Cancel"
```

Comments

If a line is too long to fit conveniently in the script window, use the character to break it into two (or more) lines for viewing.

Comments:

A line that is split with is shown in the script editor as more than one line, but when it's executed, it is treated as a single line of code.

The script editor automatically indents continued lines, as shown in the example above.

A character which is used within a literal string does not break the line, because the is treated as part of the quoted string instead of being treated as a line continuation. For example, the following statement causes a compile error because the character is inside the quotes:

```
answer "This is a test. This is only a test.
```

Had this been an actual life..." with "OK" -- BAD EXAMPLE

The above bad example can be corrected by using the && operator to break up the long string:

answer "This is a test. This is only a test."

&& "Had this been an actual life..." with "OK" -- good example

The string has been broken into two substrings, so the character is no longer within a literal string. This second example does not cause an error.

Cross-platform caution! The synonym ~ (option-L) can be used only on Mac OS and OS X systems. If you use a script containing the ~ character on a Windows or Unix system, a script error may result. To ensure cross-platform compatibility, use the synonym instead.

^

operator

Synonyms

Objects

numeric

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

* operator, exp function, exp1 function, exp10 function, exp2 function, log10 function, multiply command, sqrt function, Operator Precedence Reference

Summary

Raises a number to a power.

Syntax

number^exponent

Example Code

```
7^3 -- 7 to the 3rd power, or 343
myNumber^(1/2) -- the square root of myNumber
put 2^bitDepth into numberOfColors
```

Comments

Use the ^ operator to raise a number to a power, or to find a root of a number.

Parameters:

The number and exponent are numbers, or expressions that evaluate to numbers.

Comments:

If the exponent is a fraction, with 1 as the numerator, the ^ operator finds the specified root of the number. For example, if the exponent is 1/2, the operation yields the square root of number; if the exponent is 1/4, the operation yields the 4th root of the number, and so on.

If the exponent is zero, the result of this operator is 1, regardless of what the number is.

abbreviated

keyword

Synonyms

abbr, abbrev

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

convert command, date function, english keyword, ID property, long keyword, name property, short keyword, system keyword, time function

Summary

Specifies a format for the date and time functions, the convert command, and the name, ID, and owner properties.

Syntax

Example Code

```
the abbreviated date  
put (the abbreviated name of this card) & return after cardsList
```

Comments

Use the abbreviated keyword to obtain a date, time, name, or ID with a moderate amount of detail.

Comments:

In general, an abbreviated form is longer than the short form, but shorter than the long form.

If the useSystemDate property is set to false, an abbreviated date looks like this:

Thu, Jan 27, 2000

If the useSystemDate is true, the abbreviated date is formatted according to the system settings.

If the useSystemDate property is set to false, an abbreviated time looks like this:

11:22 AM

If the useSystemDate is true, the abbreviated time is formatted according to the system settings.

An abbreviated object name consists of the object type, followed by the name in quotes. For example, an abbreviated card name looks like this: card "This Card"

An abbreviated object ID looks like this: field ID 2238

Note: The abbreviated keyword is implemented internally as a property, and appears in the `propertyNames`. However, it cannot be used as a property in an expression, nor with the `set` command.

Changes to Transcript:

The form the abbreviated owner was introduced in version 2.0. In previous versions, the form of the owner property could not be specified and reported only the abbreviated owner.

abs

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

round function, trunc function, Recipe for an approximate-equality function

Summary

Returns the absolute value of a number.

Syntax

the abs of number
`abs(number)`

Example Code

```
abs(14) -- returns 14
the abs of -14 -- returns 14
abs(firstNum - secondNum) -- is equal to the next example
abs(secondNum - firstNum) -- is equal to the previous example
```

Comments

Use the abs function to determine the difference between two numbers when you don't know ahead of time which is greater.

Parameters:

The number is a positive or negative number, or any expression that evaluates to a number.

Value:

The abs function returns a positive number.

Comments:

The absolute value of a number is that number's distance from zero. If the number is positive, its absolute value is just the number; if the number is negative, its absolute value is the negative of the number. Because of this, the absolute value of a number is always positive.

acceleratorKey

property

Synonyms

accelKey

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

acceleratorModifiers property, acceleratorText property, commandKeyDown message, mouseUp message

Summary

Specifies a shortcut key combination for a button's mouseUp handler.

Syntax

set the accel[erator]Key of button to {empty | letterChar}

Example Code

```
set the acceleratorKey of button 1 to "a"  
set the acceleratorKey of button "Switch Order" to "F8"  
set the accelKey of button "Calc" to field "Key"
```

Comments

Use the acceleratorKey property to give users a shortcut key combination for often-used buttons, or to provide a keyboard shortcut to a button that's used as a menu item in a stack menu.

Value:

The acceleratorKey of a button is a single lowercase letter from a to z, or a key name. Setting the acceleratorKey to empty removes the shortcut.

By default, the acceleratorKey of a newly created button is empty.

Comments:

Press the key combination defined in a button's acceleratorKey property to send a mouseUp message to the button, instead of clicking.

The key(s) specified in the acceleratorMods property must be pressed along with the acceleratorKey.

Gotcha: The acceleratorKey property is case-sensitive. You must specify a lowercase letter as the letterChar; the uppercase letter is not equivalent.

If the insertion point is in a field, the keypress is sent to the field, and the button does not receive it.

On Unix systems, the key names are listed in the file `"/usr/include/X11/keysymdef.h"`. Don't include the `"XK_"` prefix in these key names; for example, use `"F8"` for the key designated as `"XK_F8"` in the file.

acceleratorModifiers

property

Synonyms

accelMods

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

acceleratorKey property, acceleratorText property, mnemonic property, mouseUp message

Summary

Specifies one or more modifier keys that must be pressed with the button shortcut key specified by the acceleratorKey property.

Syntax

set the acceleratorModifiers of button to {empty | keys}

Example Code

```
set the acceleratorModifiers of button 1 to shift
set the accelMods of button "Speak" to control,alt
set the accelMods of button it to command,shift
```

Comments

Use the acceleratorModifiers property, along with the acceleratorKey property, to specify a shortcut key combination for a button.

Value:

The acceleratorModifiers of a button consists of a list of one or more keys, separated by commas.

The keys can be any of "control", "command", "alt", "option", and "shift". If you use "command" as a key, it is converted to "control". If you use "option" as a key, it is converted to "alt".

By default, the acceleratorModifiers of a newly created button is empty.

Comments:

The following statements set up a shortcut for a button called "Calculate":

```
set the acceleratorKey of button "Calculate" to "C"
set the acceleratorModifiers of button "Calculate" to alt,shift
```

When the user presses Alt-Shift-C (on Unix or Window systems) or Option-Shift-C (on Mac OS systems), the button's mouseUp handler is executed.

Cross-platform note: On Mac OS systems, the Control key and Command key are equivalent when used with this property.

acceleratorText

property

Synonyms

accelText

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

acceleratorKey property, acceleratorModifiers property, label property, mnemonic property, rightMargin property, textAlign property

Summary

Specifies the shortcut hint that appears at the right edge of a button.

Syntax

set the acceleratorText of button to string

Example Code

```
set the acceleratorText of button 3 to "A"  
set the accelText of button "Go" to field "Shortcut Key"  
set the accelText of button 9 to "option-D"  
set the accelText of the mouseControl to empty -- remove acceleratorText
```

Comments

Use the acceleratorText property to provide the user with an onscreen hint about the shortcut key combination specified by the button's acceleratorKey property.

Value:

The acceleratorText of a button is a string, or an expression that evaluates to a string.

By default, the acceleratorText of a newly created button is empty.

Comments:

The acceleratorText string appears at the right edge of the button, inside the button's rightMargin. Revolution does not automatically make room for the acceleratorText string, so you might need to enlarge the button to prevent its label from overlapping the acceleratorText string.

If the button's textAlign property is set to "right", the button's label overlaps the acceleratorText string, so you shouldn't set a button's acceleratorText if it has a right-aligned label.

Usually, the string should be the character in `acceleratorKey`, but you might want to include additional characters to hint at the modifier key needed.

Note: Nothing happens when the user presses the key specified by the `acceleratorText` property unless the `acceleratorKey` and `acceleratorModifiers` are set. The `acceleratorText` property creates a visual hint, but does not create the actual shortcut key combination.

accentColor

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

colorNames function, hiliteColor property, lookAndFeel property, About colors and color references, Color Names Reference, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies the highlight color used for the active menu item.

Syntax

set the accentColor to {colorName | RGBColor}

Example Code

```
set the accentColor to "blue"
set the accentColor to 255,0,0 -- bright red
set the accentColor to "#CC00FF" -- violet
```

Comments

Use the accentColor to hilite a menu item associated with a button, when the user is in the process of choosing that menu item—that is, when the menu is open and the mouse pointer is over the menu item.

Value:

The accentColor is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

Comments:

This property is used only when the lookAndFeel property is set to "Macintosh" or "Windows 95". If the lookAndFeel property is set to "Appearance Manager" or "Motif", the accentColor has no effect.

The `accentColor` property is copied from the system settings every time Revolution starts up, but can be changed by a handler.

accept

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close socket command, hostNameToAddress function, open socket command, openSockets function, read from socket command, write to socket command, How to unwedge upload and download operations

Summary

Accepts an Internet connection and creates a socket for that connection.

Syntax

accept [datagram] connections on port number with message callbackMessage

Example Code

```
accept connections on port 80 with message "webConnect"  
accept datagram connections on port it with message myMessage
```

Comments

Use the accept command when running a server, to accept TCP connections or UDP datagrams from other systems (or other processes on the same system).

Use the datagram option if you want to accept UDP datagrams.

Parameters:

The portNumber is the TCP port number on which to accept connections.

The callbackMessage is the name of a message to be sent when a connection is made or a datagram is received.

Comments:

When a connection is made or a datagram is received, the accept command creates a new socket that can be used to communicate with the other system (or process). When using the close socket, read from socket, or write to socket commands, you can refer to this socket with a socket identifier that looks like this:

host:port[|connectionID]

where the connectionID is a number assigned by the accept command. (You only need to specify the connection number if there is more than one socket connected to a particular port and host.)

The callbackMessage is sent to the object whose script contains the accept command. Either one or two parameters are sent with this message. The first parameter is the IP address of the system or process making the connection. If a datagram is being accepted, the second parameter is the contents of the datagram.

For technical information about sockets, see RFC 147 at <<http://www.ietf.org/rfc/rfc147.txt>>.

For technical information about UDP datagrams, see RFC 768 at <<http://www.ietf.org/rfc/rfc0768.txt>>.

For technical information about the numbers used to designate standard ports, see the list of port numbers at <<http://www.iana.org/assignments/port-numbers>>, in particular the section entitled "Well Known Port Numbers".

acceptDrop

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

dragDrop message, dragEnd message, dragEnter message, How to prevent dragging and dropping to a field, How to respond to a drag and drop

Summary

Specifies whether a drop will be accepted during a drag and drop.

Syntax

set the acceptDrop to {true | false}

Example Code

```
set the acceptDrop to true
if word 1 of the target is "button" then set the acceptDrop to false
```

Comments

Set the acceptDrop property in an object's dragEnter handler to allow it to accept drops.

Value:

The acceptDrop is true or false.

Comments:

Usually, you set the acceptDrop property to true in a dragEnter handler to indicate that the target of the dragEnter message will accept a drop of the data being dragged.

If the acceptDrop is set to true, when you drop data, a dragDrop message is sent to the object that the mouse pointer is over, and a dragEnd message is sent to the object that was dragged from. If the acceptDrop is false, no dragDrop or dragEnd message is sent.

Revolution handles the mechanics of dragging and dropping text between and within unlocked fields. To support this type of drag and drop operation, you don't need to do any scripting. However, drag and drop of other types of data or between other object types is not automatic: setting the acceptDrop property to true does not, by itself, implement dragging and dropping. It only allows the dragDrop message to be sent.

The `acceptDrop` property is automatically set to true when a `dragEnter` message is received by an unlocked field (unless the message is trapped), and is automatically set to false when a `dragEnter` message is received by any other object. (When the `dragEnter` message reaches the engine, the engine sets the property to true if the target is an unlocked field and to false otherwise.) This allows unlocked fields to accept dragged text without requiring you to do any scripting. To prevent an unlocked field from accepting a drag, use a `dragEnter` handler and either set the `acceptDrop` to false, or trap the `dragEnter` message by not passing the message.

The `acceptDrop` is automatically set to false when the mouse pointer enters a locked field, or any other object. To allow an object other than a locked field to accept drops, set the `acceptDrop` to true in a `dragEnter` handler.

acos function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

asin function, atan function, cos function, pi constant, sin function, tan function

Summary

Returns the arc cosine of a number, in radians.

Syntax

the acos of number
`acos(number)`

Example Code

```
acos(-1) -- returns pi  
acos(cos(.3)) -- returns .3  
acos(field "Opposite" + field "Sine")
```

Comments

Use the acos function to find the arc cosine of a number.

Parameters:

The number is a number between -1 and 1, or an expression that evaluates to such a number.

Value:

The acos function returns a number between zero and pi.

Comments:

The arc cosine of number is an angle whose cosine is equal to number. In other words, acos is an inverse of the cos function.

The result of the acos function is returned in radians. To get this result in degrees, use the following function:

```
function acosInDegrees theMagnitude  
  return acos(theMagnitude) * 180 / pi
```

end acosInDegrees

activatePalettes

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

hidePalettes property, lookAndFeel property, palette command, raisePalettes property, style property

Summary

Specifies whether palettes have the same appearance when active and inactive.

Syntax

set the activatePalettes to {true | false}

Example Code

```
set the activatePalettes to false
if the activatePalettes then hideAllPalettes
```

Comments

On Mac OS systems, inactive palettes have the same appearance as the frontmost palette. Use the activatePalettes property to control this behavior.

Value:

The activatePalettes property is true or false.

By default, the activatePalettes is true if Revolution is running on a Mac OS system and false otherwise.

Comments:

If the activatePalettes is true, all palettes look the same. If it is false, the active palette window looks different from the other palette windows.

add

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

+ operator, divide command, multiply command, numberFormat property, subtract command

Summary

Adds a number to a container and places the resulting value in the container.

Syntax

add number to [chunk of] container
add {number | array} to arrayContainer

Example Code

```
add 7 to field "Previous Amount"  
add field "New" to summaryOfInventory  
add qty * price to last line of myOrder
```

Comments

Use the add command to add a number to a container or a portion of a container, or to add two arrays containing numbers.

Parameters:

The number is an expression that evaluates to a number.

The chunk is a chunk expression specifying a portion of the container.

The container is a field, button, or variable, or the message box.

The arrayContainer is an array variable each of whose elements is a number.

Comments:

The contents of the container (or the chunk of the container) must be a number or an expression that evaluates to a number.

If a number is added to an arrayContainer, the number is added to each element. If an array is added to an arrayContainer, both arrays must have the same number of elements and the same dimension, and each element in the array is added to the corresponding element of the arrayContainer.

If the container or an element of the arrayContainer is empty, the add command treats its contents as zero.

If container is a field or button, the format of the sum is determined by the numberFormat property.

Changes to Transcript:

The add to arrayContainer form was introduced in version 1.1. In previous versions, only single numbers could be used with the add command.

addMax

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of button "Click Here" to addMax
```

Comments

Use the addMax keyword to "wash out" the area under an object.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to control how the object's color is displayed. When the addMax mode is used, each component of the object color (red, green, and blue) is compared with the corresponding component of the color underneath, and the maximum of each is used for the displayed color.

For example, suppose an object's color is 30,70,150, and the color of the pixels under the object is 60,40,100. If the addMax mode is used, the object's displayed color is 60,70,150.

The addMax mode can be used only on Mac OS systems. On Unix and Windows systems, objects whose ink property is set to this mode appear as though their ink were set to srcCopy.

addOver

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

addMax keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword, Object menu > Inks

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of button 2 to addOver
```

Comments

Use the addOver keyword to combine an object's color with the colors underneath it.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to control how the object's color is displayed. When the addOver mode is used, each component of the object color (red, green, and blue) is added to the corresponding component of the color underneath. If the result is greater than 255, the component rolls over, back to zero.

For example, suppose an object's color is 25,220,150, and the color of the pixels under the object is 60,40,100. If the addOver mode is used, the object's displayed color is 85,4,250.

The addOver mode can be used only on Mac OS systems. On Unix and Windows systems, objects whose ink property is set to this mode appears as though their ink were set to srcCopy.

addPin

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

addMax keyword, addOver keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of button "Bar" to addPin
```

Comments

Use the addPin keyword to combine an object's color with the colors underneath it.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed. When the addPin mode is used, each component of the object coloróred, green, and blueóis added to the corresponding component of the color underneath. If the resulting number is greater than 127 (the maximum), 127 is used for that component.

For example, suppose an object's color is 240,0,100, and the color of the pixels under the object is 60,40,20. If the addPin mode is used, the object's displayed color is 127,40,120.

The addPin mode can be used only on Mac OS systems. On Unix and Windows systems, objects whose ink property is set to this mode appears as though their ink were set to srcCopy.

address

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hostName function, platform function, request appleEvent command, send to program command

Summary

Reports the name of the system the application is running on.

Syntax

get the address

Example Code

```
put the address into myAddress  
put last item of the address into theAppName
```

Comments

Use the address property to find out the path to the application, or the name of the system.

Value:

The address reports the computer's name, a colon, and the path to the application.

The address property is read-only and cannot be set.

Comments:

For example, if Revolution is running on a Mac OS system named "Heliand", and the Revolution folder is located on a volume called "Fnord", its address property is Heliand:/Fnord/Revolution/Revolution.

On Unix systems, the computer's name is its domain name. On Mac OS systems, the name is the one set in the File Sharing control panel.

Changes to Transcript:

Support for AppleTalk zone addresses on Mac OS systems was removed in version 1.1. In previous versions, the address property reported the application's AppleTalk address, including its zone.

adMin

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of field ID 223 to adMin
```

Comments

Use the adMin keyword to darken the area under an object.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed. When the adMin mode is used, each component of the object color (red, green, and blue) is compared with the corresponding component of the color underneath, and the minimum of each is used for the displayed color.

For example, suppose an object's color is 30,70,150, and the color of the pixels under the object is 60,40,100. If the adMin mode is used, the object's displayed color is 30,40,100.

The adMin mode can be used only on Mac OS systems. On Unix and Windows systems, objects whose ink property is set to this mode appears as though their ink were set to srcCopy.

after
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

before keyword, into keyword, put command, select command, Recipe for building a repeated string

Summary

Used with the put command to place a value at the end of a container or chunk.

Syntax

Example Code

```
select after field 1 -- puts the insertion point at the end of the field  
put space after item 2 of field "Formatted Address"
```

Comments

Use the after keyword to position the insertion point at a selected location in a field, or to add text at a specific location in a container.

Comments:

You can specify either a container, or a chunk within a container. If you don't specify a chunk, the after keyword specifies the very end of the container.

When you use the after keyword, the current contents of the container or chunk is not affected. The text is added to the container or chunk, instead of replacing it.

aliasReference

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

create alias command, defaultFolder property

Summary

Returns the name and location of the file or folder that an alias, symbolic link, or shortcut refers to.

Syntax

the aliasReference of aliasPath

aliasReference(aliasPath)

Example Code

```
the aliasReference of "/Disk/Folder/Alias"  
put the aliasReference of it into fileToOpen
```

Comments

Use the aliasReference function to perform an operation on a file after the user has selected an alias, symbolic link, or shortcut to the file.

Parameters:

The filePath is the location and name of the alias whose referenced file or folder you want to get. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

Value:

The aliasReference function returns the name and location of a file or folder.

Comments:

If the filePath does not exist, the aliasReference function returns empty and the result is set to "can't get".

allowFieldRedraw

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

lock screen command, unlock screen command

Summary

Has no effect and is included in Transcript for compatibility with imported SuperCard projects.

Syntax

set the allowFieldRedraw to {true | false}

Example Code

Comments

In SuperCard, the allowFieldRedraw property determines whether scrolling fields are redrawn after scrolling. In Revolution, scrolling fields are always redrawn.

The allowFieldRedraw property is always set to false. A handler can set it to any value without causing a script error, but the actual value is not changed.

allowInlineInput

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

lockText property, textFont property, traversalOn property, useUnicode property

Summary

Specifies whether the user can type non-Roman text directly into a field.

Syntax

set the allowInlineInput to {true | false}

Example Code

```
set the allowInlineInput to true
set the allowInlineInput to the letUsersType of this stack
```

Comments

Use the allowInlineInput property to determine the method by which the user can enter double-byte text.

Value:

The allowInput property is true or false.

By default, the allowInlineInput is true.

Comments:

If the allowInlineInput property is set to true, the user can type double-byte characters (used for languages that do not use the Roman alphabet, such as Chinese, Japanese, and Korean) directly into fields.

If the allowInlineInput is false, if the insertion point is in a field and a double-byte language is currently selected, a text-entry palette appears where the user can enter the characters. (This palette is displayed by the operating system's language software, not by Revolution.) When the user presses Enter, the characters are placed in the field.

allowInterrupts

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantAbort property, errorDialog message, interrupt function, lockErrorDialogs property, Why can't I interrupt a handler?, Shortcut to stop a running handler

Summary

Specifies whether the user can halt a handler with a key combination.

Syntax

set the allowInterrupts to {true | false}

Example Code

```
set the allowInterrupts to false
set the allowInterrupts to the hilite of button 1
```

Comments

Use the allowInterrupts property to prevent users from interrupting handlers that must run to completion. For example, some handlers that change data cannot be interrupted safely, because they will leave data in an inconsistent state if interrupted.

Value:

The allowInterrupts property is true or false.

By default, the allowInterrupts property is true.

Comments:

If the allowInterrupts property is set to true, the user can halt handlers by typing Control-period or Control-break (on Windows or Unix) or Command-period (on Mac OS). Setting this property to false disables this capability and ensures that the user cannot interrupt a handler.

If the allowInterrupts property is set to false and the user attempts to interrupt the handler, the interrupt function returns true. To provide a clean exit, check this function and do any needed cleanup tasks before exiting the handler.

Setting the `allowInterrupts` property to `false` is functionally equivalent to setting the `cantAbort` property to `true` for each open stack.

Caution! Before setting the `allowInterrupts` property to `false`, make sure all handlers that may be affected have been thoroughly tested. If `allowInterrupts` is set to `true`, you cannot interrupt a runaway handler with the standard key combination.

allowKeyInField

property

Synonyms

Objects

field, global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

enterInField message, keyDown message, returnInField message

Summary

Has no effect and is included in Transcript for compatibility with imported SuperCard projects.

Syntax

set the allowKeyInField to {true | false}

Example Code

Comments

In SuperCard, the allowKeyInField property determines whether a "keyInField" message is sent when a key is pressed while the insertion point is in a field.

In Revolution, the "keyInField" message is not sent.

The allowKeyInField property is always true. A handler can set it to any value without causing a script error, but the actual value is not changed.

alphaData

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

blendLevel property, imageData property, import command, maskData property

Summary

Specifies the binary data that makes up the alpha channel of the picture in an image object.

Syntax

set the alphaData of image to binaryData

Example Code

```
set the alphaData of image "Hellacious" to savedAlphaChannel
if charToNum(char x of the alphaData of image 1) is 255 then next repeat
```

Comments

Use the alphaData property to control the transparency of the pixels in an image.

Value:

The alphaData of an image consists of a sequence of binary values.

Comments:

Each pixel is represented by 8 bits (1 byte) of alpha channel data, with pixels numbered from the top left corner of the image, left to right, then top to bottom.

A value of zero means the pixel is fully transparent; a value of 255 is fully opaque; and values in between indicate a level of partial translucency.

Gotcha: Since the alphaData of an image is binary data rather than text, trying to display the data in a field may cause unexpected behavior.

Since each pixel is represented by 8 bits (1 byte or 1 character), you can obtain the numeric value for a given pixel using the charToNum function. For example, the numeric value of the alphaData for the tenth pixel is given by the expression charToNum(char 10 of the alphaData of image).

Gotcha: When changing the `alphaData` property, make sure the new data is the correct size: 1 byte per pixel in the image. If you set an image's `alphaData` property to data whose total length is incorrect, the image appearance may be distorted.

alternateLanguages

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

do command

Summary

Returns a list of the OSA script languages that are installed on the system.

Syntax

the alternateLanguages

alternateLanguages()

Example Code

```
the alternateLanguages
if "AppleScript" is among the lines of the alternateLanguages
    then do it as AppleScript
```

Comments

Use the alternateLanguages function to find out what scripting languages (in addition to Transcript) are available for use with the do command.

Value:

The alternateLanguages function returns a list of script languages, one per line.

Comments:

Mac OS and OS X systems support system-wide script languages (such as AppleScript) through the Open Scripting Architecture (OSA). You can write statements in any OSA language and execute them using the do command.

altID

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

ID property

Summary

Specifies an alternate ID for objects.

Syntax

set the altID of object to IDNumber

Example Code

```
put the altID of this card into myID
repeat while the altID of stack myStack <> 0
```

Comments

Use the altID property to specify an additional ID for an object. Both the ID and the altID property are checked when you refer to an object by ID.

Value:
The altID property of an object is a non-negative integer.

By default, the altID for all objects is zero.

Comments:

This property can be used to ensure compatibility with imported SuperCard and HyperCard stacks that assume buttons and fields are created with sequential IDs. You can also use the altID of a stack as a consistent way of referring to it, since the ID of a stack changes every time an object is created.

Gotcha: Be careful not to set an object's altID property to the ID of an object of the same type. Since both properties are checked when you refer to an object by ID, doing this may cause the wrong object to be found because its altID property is the same as the ID of the object you want.

altKey

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

commandKey function, controlKey function, down constant, keyDown message, keysDown function, keyUp message, metaKey function, mouse function, optionKey function, optionKeyDown message, shiftKey function, up constant

Summary

Returns the state of the Alt key.

Syntax

the altKey

altKey()

Example Code

```
put the altKey into keyState
if the altKey is down then exit mouseUp
repeat until altKey() = up
```

Comments

Use the altKey function to check whether the Alt key, Meta key, or Option key is being pressed. You can use altKey to add alternative capabilities to user actions such as clicking.

Value:

The altKey function returns down if the key is pressed and up if it's not.

Comments:

The altKey, optionKey, and metaKey functions all return the same value. Which one to use is a matter of preference.

The terminology varies depending on platform. Users of different operating systems may know this key as the Option key (Mac OS systems), Meta key (Unix systems), or Alt key (Windows systems).

alwaysBuffer property

Synonyms

Objects

image, player, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bufferHiddenImages property, pixmapID property, screenNoPixmaps property, screenSharedMemory property, windowID property, Why am I running out of memory?

Summary

Determines whether the contents of windows, players, and images are buffered offscreen.

Syntax

set the alwaysBuffer of {stack | image | player} to {true | false}

Example Code

```
set the alwaysBuffer of this stack to true
set the alwaysBuffer of image "Smile" to false
set the alwaysBuffer of the templatePlayer to true
```

Comments

Use the alwaysBuffer property to eliminate unwanted flicker when objects are being redrawn. This property is especially useful for eliminating flicker when using animation in a stack.

Value:

The alwaysBuffer of an object is true or false. By default, the alwaysBuffer property of newly created images, players, and stacks is set to false.

Comments:

When moving objects are displayed on the screen, flicker may result, since the objects are being redrawn as they move. You can avoid this problem by judicious use of the alwaysBuffer property, which creates an offscreen memory area in which objects are initially drawn. The visible display is updated from this buffer area, eliminating flicker.

Setting a stack's alwaysBuffer property to true eliminates flicker, but increases memory usage because memory must be allocated for the buffer. To make most efficient use of memory, Revolution automatically buffers stacks under the following circumstances:

- visual effects are being executed

- the move command is operating
- objects are selected

The alwaysBuffer property overrides this behavior and ensures that the stack is always buffered.

You can find out whether a stack is currently being buffered by comparing its pixmapID property to its windowID property: if these two properties are not the same, the stack is buffered.

Setting an image's alwaysBuffer property to true forces the image to uncompress immediately, even if the image is hidden. This speeds up using the show command to display an image. Setting the alwaysBuffer property of all images to true is equivalent to setting the global bufferHiddenImages property to true.

Setting a player's alwaysBuffer property to true forces the movie to be drawn in an offscreen buffer. This prevents the movie from flickering when other objects (such as buttons) are drawn on top of it. It also allows the current frame to be seen when the card is printed.

If a player's movie contains only sound with no visual track, the setting of its alwaysBuffer property has no effect.

If a player's alwaysBuffer is false, the movie it contains is drawn in front of all objects. The visual effect command does not affect the screen area inside the rectangle of a player whose alwaysBuffer is false. If a player's alwaysBuffer is true, it cannot be controlled with the controller bar and must be operated by script control.

Note: Setting a player's alwaysBuffer to true always increases memory usage, and may make movie playing more jerky.

and

operator

Synonyms

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bitAnd operator, false constant, not operator, or operator, true constant, Operator Precedence Reference

Summary

Evaluates to true if both operands are true, false otherwise.

Syntax

value1 and value2

Example Code

```
(1 > 0) and (1 = 0) -- evaluates to false
(1 > 0) and (1 = 1) and (0 = 0) -- evaluates to true
if the shiftKey is down and myCount > 1 then exit mouseUp
```

Comments

Use the and operator to combine two or more logical values.

Parameters:

The value1 and value2 are true or false, or expressions that evaluate to true or false.

Comments:

If value1 is false or value2 is false, or if both value1 and value2 are false, then the and operation evaluates to false. If value1 and value2 are both true, the expression value1 and value2 evaluates to true.

You can combine the logical operators and, or, and not in an expression.

Note: Transcript uses what is known as "short-circuit evaluation" for logical operators. This means that value1 is evaluated first. If value1 is false, the expression value1 and value2 is false regardless of what value2 is (because the expression evaluates to false unless both the values are true). In this case, Revolution does not evaluate value2, since doing so is not necessary to determine the value of value1 or value2. For example, evaluating the expression asin(2) normally causes an execution error (because 2 is not a legal argument for the arc sine function), but evaluating the expression (1 = 0) and (asin(2) = 1)

does not cause an error: since $(1 = 1)$ is always false, the whole statement is always false and Revolution never tries to evaluate the `asin` function.

angle

property

Synonyms

Objects

graphic, image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

arcAngle property, polySides property, revRotatePoly command, rotate command, startAngle property

Summary

Specifies the starting angle of an arc or the rotation angle of a regular polygon or image.

Syntax

set the angle of {graphic | image} to angleInDegrees

Example Code

```
set the angle of graphic "Triangle" to 180
set the angle of image ID 45902 to it
set the angle of graphic myGraphic to the angle of myGraphic + 5
```

Comments

Use the angle property to create an arc shape, or to turn a regular polygon or image object.

Value:

The angle of a graphic or image is an integer between zero and 360.

By default, the angle property of a newly created graphic or image is zero.

Comments:

If you specify an angleInDegrees greater than 360, the angle is set to the number you specify mod 360.

A regular polygon can be rotated between zero and 360°. Increasing the angleInDegrees rotates the polygon clockwise. You can specify the angle of a graphic that is not a regular polygon, but it has no effect on the graphic's appearance.

An image can be rotated between zero and 360°. Increasing the angleInDegrees rotates the image counterclockwise. Unlike the rotate command, the angle property affects only the screen display of the image, not the actual picture data in it. Changing an image's angle does not change the imageData of the image. Repeated changes to the angle property, unlike repeated uses of the rotate command, do not

degrade the image's quality. The rotate command cannot be used on a referenced image, but the angle of a referenced image can be set.

By default, oval graphics display their entire arc from zero to 360°, forming a complete oval. You can use the angle and arcAngle properties to specify a portion of the oval to be displayed. The angle property determines the starting point of the arc. Zero is at the right edge, 3 o'clock. Increasing the angle moves the starting point counter-clockwise around the arc. (The direction of rotation for an arc is opposite the direction of rotation for a polygon.) For example, if the angle is 90, the arc starts at the top edge of the graphic's rectangle, 12 o'clock.

Note: For an oval, the angle is the same as the startAngle. Changing one changes the other.

Changes to Transcript:

The ability to set the angle of an image was introduced in version 2.0. In previous versions, the angle property applied only to graphics.

annuity function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

compound function, round function

Summary

Computes the value of an annuity given an interest rate and a number of payments.

Syntax

`annuity(interestRate,numberOfPeriods)`

Example Code

```
annuity(.08,10)  
annuity(currentAnnualRate/12,monthsOfLoan)
```

Comments

Use the annuity function to calculate the present or future value of an annuity or to calculate loan payments.

Parameters:

The interestRate is a positive number. The interestRate is expressed as a fraction of 1 so, for example, an 8% rate is written .08.

The numberOfPeriods is a positive number.

Value:

The annuity function returns a positive number.

Comments:

The formula for the value of an ordinary annuity is
$$(1 - (1 + \text{interestRate})^{-(\text{numberOfPeriods})})/\text{interestRate}$$

The annuity function calculates this value.

The numberOfPeriods and the interestRate must use the same unit of time. For example, if the periods are months, the interest rate is the interest per month.

You can use the annuity function to calculate the amount of loan payments as follows:

$$\text{paymentAmount} = \text{totalAmount} / \text{annuity}(\text{rate}, \text{periods})$$

For example, if the loan is for \$2500 at an interest rate of 2% per month and is to be repaid in a year, the monthly payment is $2500 / \text{annuity}(.02, 12)$ or \$236.40.

answer
command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

answer color command, answer effect command, answer file command, answer folder command, answer printer command, ask command, gRevAppIcon keyword, gRevSmallAppIcon keyword, it keyword, modal command, Why doesn't the ask or answer dialog appear in my standalone?, Recipe for Hello World, Recipe for speaking an alert message

Summary

Displays a dialog box with a message and up to seven buttons.

Syntax

```
answer [iconType] prompt [with button1 [or buttons]]  
  
[titled windowTitle] [as sheet]
```

Example Code

```
answer "Go ahead?" with "Yes" or "No" or "Maybe"  
answer information filesProcessed && "files were exported."  
answer "Greetings!" with "Log In" or "Cancel" titled "Login"  
answer the currPrompt of me with "OK" or "Cancel" as sheet
```

Comments

Use the answer command to get information or a confirmation from the user before continuing. The user must click one of the buttons to dismiss the dialog box.

Parameters:

The iconType parameter is one of the following types: information, question, error, or warning. The icon is displayed on the left side of the dialog box. If you don't specify an icon, none is displayed.

Cross-platform note: On OS X systems, the image specified by the gRevAppIcon keyword appears if you don't specify an iconType. If you specify an iconType, the image specified by the gRevSmallAppIcon keyword appears instead, along with the standard icon specified by the iconType.

The prompt is a string (or any expression that evaluates to a string). The dialog box expands if necessary to fit the contents.

The buttons are strings. You can specify up to seven buttons, separated by the word "or". The buttons and the dialog box expand if necessary to fit the button names; the total number of characters in the buttons is limited only by the maximum dialog box size and the font size. The last button you specify is the default button. (Pressing Return or Enter is equivalent to clicking the default button.) If you don't specify any button names, the dialog box contains a single OK button.

The windowTitle, if specified, appears in the title bar of the dialog box. If you don't specify a windowTitle, the title bar is blank.

Comments:

The prompt can be either formatted text (in the htmlText property's format) or plain text. If the prompt contains <p> or a start/end tag pair, the answer command assumes the text is in the same format as the htmlText property. Otherwise, the answer command assumes the text is plain text.

The name of the button the user chooses is placed in the it variable.

The position and appearance of the dialog box varies between platforms. On Mac OS systems, the dialog box is centered on the screen; on Unix and Windows systems, the dialog box is centered over the active window. On Windows systems, the buttons are shown in reverse order (the first button is on the right side).

On OS X systems, the image specified in the gRevAppIcon variable appears as the application icon in the answer dialog box (unless the answer...as sheet form is used). If you specify an iconType, the image specified in the gRevSmallAppIcon variable is used instead, along with the special icon specified by the iconType.

If the as sheet form is used, the dialog box appears as a sheet on OS X systems. On other systems, the as sheet form has no effect and the dialog box appears normally. Attempting to open a sheet from within another sheet displays the second stack as a modal dialog box instead.

Changes to Transcript:

The answer...as sheet form was introduced in version 2.0.

The ability to provide formatted text for the prompt was introduced in version 2.0.

answer color

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

answer command, backgroundColor property, colorNames function, colors property, foregroundColor property, it keyword, About colors and color references, Color Names Reference, Recipe for translating a color name to an RGB numeric triplet, Text menu > Color

Summary

Displays the operating system's standard color-selection dialog box.

Syntax

answer color [with startingColor]

Example Code

```
answer color
if theItem is "Custom Color..." then answer color
answer color with "#FF0033"
answer color with "AliceBlue"
```

Comments

Use the answer color command to select a custom color.

Parameters:

The startingColor is a color reference consisting of one of the following:

- a standard color name
- three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue
- an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

Comments:

The answer color command displays a dialog box where the user can select a color. (This dialog box is displayed by the operating system, not by Revolution.)

The color the user chooses is placed in the it variable. If the user cancels the dialog, the it variable is set to empty, and the result function returns "Cancel".

If you specify a `startingColor`, the dialog box displays that color by default.

The color is returned in the form of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue. This format can be used directly to set any color property.

Changes to Transcript:

The option to specify a `startingColor` was introduced in version 1.1.1. In previous versions, the dialog box displayed white by default.

answer effect

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer command, answer record command, hide command, QTEffects function, show command, visual effect command, How to find out whether QuickTime is available

Summary

Displays the QuickTime special effects dialog box.

Syntax

answer effect

Example Code

```
answer effect
if the altKey is down then answer effect
```

Comments

Use the answer effect command to choose a visual effect, or to set up an effect with complex options for later use.

Comments:

The answer effect command displays a dialog box where the user can select a visual effect and (for some effects) set parameters such as speed and direction. (This dialog box is displayed by QuickTime, not by Revolution.)

An encoded description of the visual effect the user chooses is placed in the it variable. You can either use the encoded description immediately, or store it (for example, in a variable or in a custom property) for later use. To display the effect, use the encoded description with the visual effect command, with the unlock screen command, or with the hide with visual effect or show with visual effect form of the hide or show command.

If the user cancels the dialog, the it variable is set to empty, and the result function returns "Cancel".

answer file

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

answer command, answer folder command, ask file command, files function, it keyword, revMacFromUnixPath function, revUnixFromMacPath function, systemFileSelector property, About filename specifications and file paths, How to list the contents of a folder, Why can't Revolution find a file I specified?, Why doesn't Revolution recognize a stack file?

Summary

Displays a standard file dialog for the user to select a file.

Syntax

answer file prompt [with defaultPath] [{with filter | of type} types]

[titled windowTitle] [as sheet]

Example Code

```
answer file "Select a file to delete:"  
answer file "Input:" with "/Macintosh HD/"  
answer file (field "Prompt") of type "RSTK" -- shows stacks  
answer file empty with filter "JPEGs,*.jpg" -- shows JPEGs
```

Comments

Use the answer file command when a handler needs the file path of a file before continuing.

Parameters:

The prompt is a string (or any expression that evaluates to a string). If you specify empty, no prompt appears.

The defaultPath is the name and location of the folder whose contents are listed when the dialog box appears. If no defaultPath is specified, the dialog box lists the contents of the last folder you used with a file dialog box.

The windowTitle, if specified, appears in the title bar of the dialog box. If no windowTitle is given, the title bar is blank. (This parameter has no effect on Mac OS systems, because Mac OS file dialog boxes don't have a title bar.)

Use the types parameter to specify which files should appear and be available for selection.

Comments:

The dialog box displayed is the same one most programs use for the "Open" command in the File menu.

The absolute file path of the file the user chose is placed in the it variable. If the user cancels the dialog, the it variable is set to empty, and the result function returns "Cancel".

Important! The answer file command does not open the file. It only displays the dialog box and retrieves the path to the file the user specifies.

If the as sheet form is used, the dialog box appears as a sheet on OS X systems. On other systems, the as sheet form has no effect and the dialog box appears normally. Attempting to open a sheet from within another sheet displays the second stack as a modal dialog box instead.

If the systemFileSelector property is set to false, Revolution's built-in dialog box is used instead of the operating system's standard file dialog.

The way file types are specified depends on the platform:

Filtering for Mac OS: You can use either with filter or of type. With either form, the types parameter consists of one or more 4-character file types, concatenated. For example, to display text and PICT-format files but no others, use a types parameter of "TEXTPICT". To display applications only, use a types parameter of "APPL".

If the types string is shorter than four characters (on a Mac OS system), the files are not filtered: all files appear in the dialog box.

Filtering for OS X: Works the same as filtering for Mac OS, except that if one of the file types is "APPL", the dialog box displays application bundles as well as single-file applications.

Filtering for Unix: You must use the with filter form if you want to specify one or more types: the of type form can't be used on Unix systems. The types parameter consists of a single wildcard expression.

Filtering for Windows: You must use the with filter form to specify one or more types: the of type form can't be used on Windows systems. A file type consists of an optional description, a line feed or comma, and one or more file extension specifications:

[description], *.extension [: *.extension...]

For example, to specify that only Revolution files should appear in the dialog box, use this statement. The description "Revolution files" is visible to the user at the bottom of the dialog, and only files with the extension ".rev" are shown:

```
answer file myPrompt with filter "Revolution files,*.rev"
```

You can specify more than one file extension for a single description. For example, to specify that only JPEG, GIF, and PNG files should appear, use this statement:

answer file myPrompt with filter "Web Graphics,* .jpg;* .gif;* .png"

You can include several descriptions, along with their file types, by separating them with commas. The descriptions appear in the list at the bottom of the file dialog box. For example, if you want to display all text and Microsoft Word files, use this command:

answer file myPrompt with filter "Text files,* .txt,MS Word files,* .doc"

Tip: If a set of filters is complex or if you use them in several places, it may be easier to put them into a variable, then use that variable in the answer file command.

Changes to Transcript:

The ability to use answer file...of type "APPL" to designate an OS X application bundle was introduced in version 2.0. In previous versions, the APPL file type showed only atomic files of type "APPL".

The answer file...as sheet form was introduced in version 2.0.

answer folder

command

Synonyms

answer directory

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer command, answer file command, ask file command, it keyword, systemFileSelector property, About filename specifications and file paths

Summary

Displays a standard file dialog for the user to choose a folder.

Syntax

answer folder prompt [with defaultPath] [as sheet]

Example Code

```
answer folder "Please choose a folder:"  
if it is empty then answer folder chooseMsg as sheet  
answer folder "Where is the data?" with "../data_stacks/recent"
```

Comments

Use the answer folder command when you want the user to choose a folder—for example, as a destination for exported files.

Parameters:

The prompt is a string (or any expression that evaluates to a string). If you specify empty, no prompt appears.

The defaultPath is the name and location of the folder whose contents are listed when the dialog box appears. If no defaultPath is specified, the dialog box lists the contents of the last folder you used with a file dialog box.

Comments:

The absolute file path of the folder the user selects is placed in the it variable. If the user cancels the dialog, the it variable is set to empty, and the result function returns "cancel".

If the as sheet form is used, the dialog box appears as a sheet on OS X systems. On other systems, the as sheet form has no effect and the dialog box appears normally. Attempting to open a sheet from within another sheet displays the second stack as a modal dialog box instead.

If the systemFileSelector property is set to false, Revolution's built-in dialog box is used instead of the operating system's standard file dialog.

Changes to Transcript:

The answer folder...as sheet form was introduced in version 2.0.

answer printer command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

Note: this command works differently across platforms.

See Also:

answer command, open printing command, print command, printMargins property, printRotated property, revShowPrintDialog command, How to display the print settings dialog box, How to print in landscape mode, File menu > Page Setup...

Summary

Displays the operating system's standard Print dialog box.

Syntax

answer printer

Example Code

```
answer printer  
if the commandKey is down then answer printer
```

Comments

Use the answer printer command to set standard printing options for later printing.

Comments:

The answer printer command displays a dialog box where the user can set certain print-related properties. (This dialog is displayed by the operating system, not by Revolution.)

The exact options are determined by the printer driver and the operating system.

Cross-platform note: On Mac OS systems, the answer printer command displays the Page Setup dialog. You can display the standard Print dialog on a Mac OS system using open printing with dialog.

answer record

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer command, answer effect command, dontUseQT property, play command, record sound command, recordChannels property, recordCompression property, recordFormat property, recordRate property, recordSampleSize property, How to find out whether QuickTime is available

Summary

Displays the QuickTime sound-recording settings dialog box.

Syntax

answer record

Example Code

```
answer record  
if the alreadyHaveSettings of this card is false then answer record
```

Comments

Use the answer record command to specify settings for use with the record sound command.

Comments:

The answer record command displays a dialog box where the user can select a sound compression format, rate, sampling rate, and whether to record in mono or stereo. Depending on the compression format, other options may also be offered. (This dialog box is displayed by QuickTime, not by Revolution.) The settings the user chooses will be used the next time the record sound command is used.

The answer record command sets the recordCompression, recordRate, recordSampleSize, and recordChannels properties in accordance with the settings chosen in the dialog box. To save the settings you choose in the dialog box and use them later without re-displaying the dialog, save the values of these properties, then restore them when you want to record sound.

If the user cancels the dialog, the result function returns "Cancel", and the recording-related properties are unchanged.

If the dontUseQT property is set to true, the answer record command cannot be used, and the result returns "could not initialize quicktime".

any
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

first keyword, last keyword, middle keyword, number property, random function, randomSeed property, How to shuffle the items or lines in a container, Recipe for an Elizabethan insult generator

Summary

Designates a randomly chosen member of a set.

Syntax

Example Code

```
put any word of field "Dictionary" into wordOfTheDay  
go to any card -- goes to a random card in the stack  
select any button
```

Comments

Use the any keyword to specify a random object of a specified type, or to designate a random chunk in a chunk expression.

Comments:

The any keyword does not examine every member of a set; it specifies just one member of the set, randomly chosen. For example, the expression

if myString is in any field
checks whether myString is in a randomly chosen field, not whether myString is somewhere in one of the fields on the card.

appleEvent

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

address property, open process command, reply command, request appleEvent command, request command, send to program command, signal message, How to respond to double-clicking a file for Mac OS or OS X

Summary

Sent to the current card whenever the application receives an Apple event.

Syntax

appleEvent class,ID,sender

Example Code

```
on appleEvent theClass,theID -- execute a set of statements in a text file
  if theClass is "misc" and theID is "doscd" then
    request appleEvent data -- get the content of the AppleEvent
    do URL ("file:" & it)
  end if
  pass appleEvent
end appleEvent
```

Comments

Handle the appleEvent message to respond to a custom Apple event, or one that you want to handle specially.

Parameters:

The class and ID together identify the exact Apple event that was received.

The class parameter is the event class: possible classes include aevt (for required Apple Events such as "open document" and "print"), misc (for miscellaneous events such as "do script"), and others.

The ID parameter is the name of the specific Apple event.

The sender parameter is the address of the process that sent the Apple event.

Comments:

Use the request `appleEvent` command to obtain the data associated with an Apple event.

For more information about Apple events, see Apple Computer's technical documentation, Inside Macintosh: Interapplication Communication, located at [<http://developer.apple.com/techpubs/mac/IAC/IAC-2.html>](http://developer.apple.com/techpubs/mac/IAC/IAC-2.html).

arcAngle

property

Synonyms

Objects

graphic, global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

angle property, startAngle property

Summary

Specifies the angle used to draw an arc.

Syntax

set the arcAngle [of graphic] to angleInDegrees

Example Code

```
set the arcAngle to myAngle -- affects ovals drawn with the paint tools
set the arcAngle of graphic "Over" to theAngle
```

Comments

Use the arcAngle property to create an arc shape from an oval graphic, or to cause the Oval paint tool to draw arcs.

Value:

The arcAngle of a graphic is an integer between zero and 360.

By default, the arcAngle property of a newly created graphic is 360.

Comments:

By default, ovals display their entire arc from zero to 360°, forming a complete oval. Use the startAngle and arcAngle properties to specify that only a portion of the oval, forming an arc, should be drawn.

The arcAngle determines how much of the oval is used to form the arc, from zero (no arc) to 360 (a full oval). For example, if the angleInDegrees is 90, a quarter-oval arc is displayed.

The global setting of the arcAngle property controls the appearance of arcs drawn with the paint tools. Once a paint arc is drawn, its appearance cannot be changed by changing the global arcAngle property.

armBorder

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

armFill property, autoArm property, borderColor property

Summary

Determines whether a button is drawn with a border when armed.

Syntax

set the armBorder of button to {true | false}

Example Code

```
set the armBorder of button "Styles" to true
set the armBorder of button nextButton to false
if the armBorder of button ID 3 then add 1 to theCounter
```

Comments

Use the armBorder property to control the appearance of an armed button.

Comments:

The color and pattern of the border are specified by the button's borderColor and borderPattern properties.

If the button's threeD or border property is true, the armBorder property has no effect and the border does not appear.

armed

property

Synonyms
arm

Objects
button

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
accentColor property, armBorder property, armFill property, autoArm property

Summary
Specifies whether a button has a changed (active) appearance due to the mouse pointer moving into it

Syntax
set the armed of button to {true | false}

Example Code
set the armed of first button to false
if the armed of button "Font" then showFontSamples

Comments
Check the armed property to determine whether a button's menu is open—that is, whether the user is in the process of choosing an item from the menu. You can also set a button's armed property to change its appearance.

Comments:
A button's armed property is analogous to its hilite property. A button is usually armed when the mouse pointer moves into it. You can automate this behavior by setting the button's autoArm property to true.

The appearance of an armed button is determined by its armBorder and armFill properties.

Changes to Transcript:
The armed keyword was introduced in version 1.1. In previous versions, the arm synonym was used.

armedIcon

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

autoArm property, icon property, showIcon property, Why do icons disappear from a standalone application?

Summary

Specifies an image to display in a button when the mouse pointer enters it.

Syntax

set the armedIcon of button to {imageID | imageName}

Example Code

```
set the armedIcon of button "Trigger" to 2245
```

Comments

Use the armedIcon property to change a button's appearance when it is armed.

Value:

The armedIcon property is the ID or name of an image to use for an icon. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the armedIcon property of newly created buttons is set to zero.

Comments:

If the button's autoArm property is false, the setting of the armedIcon property has no effect.

armFill

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

accentColor property, armed property, armBorder property, autoArm property, hiliteColor property

Summary

Determines whether a button menu is drawn with a border when armed.

Syntax

set the armFill of button to {true | false}

Example Code

```
set the armFill of button "Go Menu" to true
```

Comments

Use the armFill property to control the appearance of an armed menu item.

Value:

The armFill is true or false.

By default, the armFill property of newly created buttons is set to false.

Comments:

When armed, the button is filled with the color specified by the accentColor property.

If the button's style property is not set to "menu", you can set the button's armFill property, but it has no effect.

If the button's showBorder property is true, the armFill property has no effect and the button's fill color does not change when the button is armed.

arrow
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, cursor property, one constant

Summary

Equivalent to the number 1.

Syntax

Example Code

```
set the cursor to arrow
```

Comments

Use the arrow constant to set the cursor to an arrow shape.

Comments:

The following two statements are equivalent:

```
set the cursor to arrow  
set the cursor to 1
```

However, the first is easier to read and understand in Transcript code.

Important! If you use the arrow cursor or other standard cursors in your application, you must include the cursors when you create your standalone. Make sure the "Cursors" option on the Inclusions tab in Step 3 of the Distribution Builder is checked.

arrowKey

message

Synonyms

Objects

button, field, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

keyDown message, navigationArrows property, selectionChanged message, textArrows property, How to block or change the action of arrow keys

Summary

Sent to the active (focused) control, or to the current card if no control is focused, when the user presses an arrow key.

Syntax

arrowKey {up | down | left | right}

Example Code

```
on arrowKey theKey -- make Up arrow go to the first card
  if theKey is "up" then go to card 1
  else pass arrowKey
end arrowKey
```

Comments

Handle the arrowKey message if you want to do something special when the user presses an arrow key.

Parameters:

The parameter indicates which arrow key was pressed.

Comments:

If the arrowKey handler does not pass the message or send it to a further object in the message path, Revolution does not perform the usual arrow key action (moving the insertion point, moving the selected object, or navigating to another card). Passing the message lets the arrow key action take place.

If the focused control is part of a group, and the group's tabGroupBehavior property is set to true, no arrowKey message is sent when the user presses an arrow key.

Note: Sending an arrowKey message does nothing unless there is an arrowKey handler in the message path. This is a difference from HyperTalk's handling of the arrowKey message.

arrowSize

property

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

endArrow property, lineSize property, startArrow property

Summary

Specifies the size of an arrow at one end of a line or polygon graphic.

Syntax

set the arrowSize of graphic to size

Example Code

```
set the arrowSize of last graphic to 20
```

Comments

Use the arrowSize property to change the size of arrows at the ends of lines and polygons.

Value:

The arrowSize of a graphic is a number between 1 and 65535.

By default, the arrowSize property of newly created graphics is set to 3.

Comments:

The arrow of a graphic is proportional to the graphic's lineSize; changing the lineSize increases or decreases the size of the arrow, although the graphic's arrowSize property is not changed. If the lineSize is zero, the arrow is not visible.

Set the startArrow or endArrow properties to true to make the arrows visible.

You can set the arrowSize of graphics other than lines and polygons, but doing so has no effect.

as

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

do command, export command, go command, save command

Summary

Used with the do command to specify a scripting language; used with the export command to specify the file format to export to; used with the go command to specify the mode of a stack; used with the save command to specify the file name and location.

Syntax

Example Code

```
do field 1 as Javascript
export me to file myFile as "PNG"
go stack "Controls" as palette
save this stack as "../Backups/Current.rev"
```

Comments

Use the as keyword to modify the do, export, go, or save commands.

Comments:

Use the alternateLanguages function to find out which languages can be used with the do command on Mac OS and OS X systems.

ascending
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

descending keyword, sort command

Summary

Used with the sort command to specify that sorting is in normal order (from less to greater).

Syntax

Example Code

```
sort this stack ascending by field "Order Number"
```

Comments

Use the ascending keyword to improve the clarity of your code. For example, if a handler contains several sort commands and some are in descending order, you can use the ascending keyword explicitly to point up the fact that there are both ascending and descending sorts.

Comments:

Since ascending is the default sort order, you never actually need to use the ascending keyword; if you leave it out, the sort is performed in ascending order anyway.

asin function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

acos function, atan function, atan2 function, sin function

Summary

Returns the arc sine of a number in radians.

Syntax

the asin of number
`asin(number)`

Example Code

```
the asin of .225  
sin(asin(.5)) -- returns .5  
asin(pi/4 - sin(myAngle))
```

Comments

Use the asin function to find the arc sine of a number.

Parameters:

The number is a number between -1 and 1, or an expression that evaluates to such a number.

Value:

The asin function returns a number between $-\pi/2$ and $\pi/2$.

Comments:

The arc sine of number is an angle whose sine is equal to number. In other words, asin is an inverse of the sin function.

The result of the asin function is returned in radians. To get this result in degrees, use the following custom function:

```
function asinInDegrees theMagnitude  
    return asin(theMagnitude) * 180 / pi
```

end asinInDegrees

ask
command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

answer command, ask file command, ask password command, modal command, Why doesn't the ask or answer dialog appear in my standalone?

Summary

Displays a dialog box with a question, a text box for the user to enter a response, and OK and Cancel buttons.

Syntax

`ask [iconType] question [with defaultResponse] [titled windowTitle]`

`[as sheet]`

Example Code

```
ask "What is your name?"  
ask "Please enter your occupation:" with "Geek"  
ask field "Prompt" with firstGuess titled "Guess!"  
ask myPrompt as sheet
```

Comments

Use the ask command when a handler needs to get information from the user before continuing.

Parameters:

The iconType parameter is one of the following types: information, question, error, or warning. The icon is displayed on the left side of the dialog box. If you don't specify an icon, none is displayed.

The question is a string (or any expression that evaluates to a string).

The defaultResponse is a string, and is placed in the text box when the dialog box appears. If no defaultResponse is specified, the text box is empty when the dialog box appears.

The windowTitle appears in the title bar of the dialog box. If no windowTitle is given, the title bar is blank.

Comments:

The contents of the text box is placed in the it variable. If the user cancels the dialog, the it variable is set to empty and the result function returns "cancel".

If the ask...as sheet form is used, the dialog box appears as a sheet on OS X systems. On other systems, the as sheet form has no effect and the dialog box appears normally. Attempting to open a sheet from within another sheet displays the second stack as a modal dialog box instead.

Changes to Transcript:

The ability to specify an iconType was added in version 2.0. In previous versions, no icon was displayed.

The ask...as sheet form was introduced in version 2.0.

The ability to provide formatted text for the prompt was introduced in version 2.0.

ask file

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

answer file command, answer folder command, ask command, dontUseNS property, it keyword, modal command, systemFileSelector property, Why can't Revolution find a file I specified?

Summary

Displays a standard Save dialog for the user to enter a file name and specify a location.

Syntax

ask file prompt [with defaultFilePath] [with filter types] [as sheet]

Example Code

```
ask file "Please name the file:"  
ask file "Save data as" with "/HD/Data Stacks/filedata.rev" as sheet  
ask file empty with "Untitled" -- no prompt  
ask file "Export picture as:" with filter "JPEG file,*.jpg"
```

Comments

Use the ask file command to let the user provide the name and location of a new file.

Parameters:

The prompt is a string (or any expression that evaluates to a string). If you specify empty, no prompt appears.

The defaultFilePath consists of a folder path, or a suggested file name, or both. The filename is the portion of the path after the last slash character (/). If a folder path is provided in the defaultFilePath, the dialog box shows the contents of that folder. Otherwise, it shows the contents of the last folder you used with a file dialog box. If a suggested file name is provided in the defaultFilePath, it appears in the file name box.

The types parameter specifies which extensions are available at the bottom of the dialog box. A file type consists of an optional description, a line feed or comma, and a file extension specification:

[description],*.extension

You can include more than one file type by separating the file types with commas.

Cross-platform note: The types parameter applies only to Windows systems. On Mac OS and Unix systems, this parameter has no effect.

Comments:

The dialog box is the same one most applications use for the "Save" command in the File menu. (If the systemFileSelector property is set to true on Mac OS and Windows systems, and always on Unix systems, the application displays its own built-in dialog box, instead of the one provided by the operating system.)

The absolute file path of the file the user chose is placed in the it variable. If the user cancels the dialog, the it variable is set to empty, and the result function returns cancel.

Important! The ask file command does not create the file. It only displays the dialog box and retrieves the path to the file the user specifies.

If the as sheet form is used, the dialog box appears as a sheet on OS X systems. On other systems, the as sheet form has no effect and the dialog box appears normally. Attempting to open a sheet from within another sheet displays the second stack as a modal dialog box instead.

If the systemFileSelector property is set to false, Revolution's built-in dialog box is used instead of the operating system's standard file dialog.

Cross-platform note: On Mac OS systems where Navigation Services is in use, if you don't specify a default file name, the file name box contains "untitled". On Unix and Windows systems, the file name box is empty when the dialog box appears.

Changes to Transcript:

The ask file...as sheet form was introduced in version 2.0.

ask password

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

ask command, modal command, Why doesn't the ask or answer dialog appear in my standalone?

Summary

Displays a dialog box like the ask command, but with the characters the user types displayed as asterisks (*) for privacy.

Syntax

ask password [clear] question [with defaultAnswer]

[titled windowTitle] [as sheet]

Example Code

```
ask password clear "Password for the remote server:"  
ask password "Please log in." titled "Millenium Group Intranet"  
ask password empty with savedPassword as sheet
```

Comments

Use the ask password command to provide privacy for the information the user types. For example, if the user is in a public place, the information might be seen by someone looking over his or her shoulder. This command encrypts the text the user types, so you can also use ask password to implement a secret password.

Parameters:

The prompt is a string (or any expression that evaluates to a string).

The defaultResponse is placed in the text box when the dialog box appears. If no defaultResponse is specified, the text box is empty when the dialog box appears.

The windowTitle, if specified, appears in the title bar of the dialog box. If no windowTitle is given, the title bar is blank.

Comments:

The encrypted contents of the text box is placed in the it variable. If the user cancels the dialog, the it variable is set to empty, and the result function returns "cancel".

If the ask password clear form is used, the text box is not encrypted, and the it variable contains whatever the user entered.

If the ask password...as sheet form is used, the dialog box appears as a sheet on OS X systems. On other systems, the as sheet form has no effect and the dialog box appears normally. Attempting to open a sheet from within another sheet displays the second stack as a modal dialog box instead.

Changes to Transcript:

The ask password...as sheet form was introduced in version 2.0.

at
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

after keyword, before keyword, put command

Summary

Used with the click, play, and prepare commands to specify a point on the screen. Also used with the read from file, read from process, and write to file commands to specify where to begin reading or writing.

Syntax

Example Code

```
click at 140,200  
play videoClip "Movie" at the clickLoc  
read from file myFile at numOfChars until end
```

Comments

Use the at keyword to specify where to click or where to start reading or writing.

Comments:

A point consists of two numbers: a vertical and horizontal distance from the top left of the current stack, separated by a comma.

atan

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

acos function, asin function, atan2 function, pi constant, tan function

Summary

Returns the arc tangent of a number in radians.

Syntax

the atan of number
`atan(number)`

Example Code

```
atan(0)  
atan(tan(-1)) -- returns -1  
the atan of vectorMagnitude
```

Comments

Use the atan function to find the arc tangent of a number.

Parameters:

The number is a positive or negative number, or an expression that evaluates to a number.

Value:

The atan function returns a number between $-\pi/2$ and $\pi/2$.

Comments:

The arc tangent of number is an angle whose tangent is equal to number. In other words, atan is an inverse of the tan function.

The result of the atan function is returned in radians. To get this result in degrees, use the following custom function:

```
function atanInDegrees theMagnitude  
    return atan(theMagnitude) * 180 / pi
```

end atanInDegrees

atan2

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

acos function, asin function, atan function, pi constant, tan function

Summary

Returns the arc tangent of one number divided by another, using the sign of both.

Syntax

`atan2(yCoordinate,xCoordinate)`

Example Code

```
atan2(-1,-1) -- returns 4/3 * pi  
atan2(thisNumber,thatNumber)
```

Comments

Use the atan2 function to find the arc tangent of one number divided by another when sign is significant.

Parameters:

The yCoordinate is a number or an expression that evaluates to a number.

The xCoordinate is a number or an expression that evaluates to a number.

Value:

The atan2 function returns a number between $-\pi$ and π .

Comments:

In most cases, `atan2(y,x)` is equal to `atan(y/x)`. However, if both x and y are negative, the sign of x/y is positive. In this case, the atan function returns an angle in the first quadrant, but the atan2 function returns an angle in the third quadrant.

The result of the atan2 function is returned in radians. To get this result in degrees, use the following custom function:

```
function atan2InDegrees firstArg,secondArg
```

```
    return atan2(firstArg,secondArg) * 180 / pi  
end atan2InDegrees
```

audioClip

object

Synonyms

ac

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

templateAudioClip keyword, About object types and object references, How to list the audio clips and video clips in a stack, File menu > Import As Control > Audio File...

Summary

An object type that contains sound data.

Syntax

Example Code

```
play pause audioClip theCurrentSoundtrack
```

Comments

Use the audioClip object type to play a sound that is stored in the stack, rather than in another file.

Comments:

Unlike a player, an audio clip contains the sound that it plays. This increases the memory required by your stack, because the sound data is loaded into memory along with the rest of the stack whenever the stack file is open. However, it prevents the sound from being accidentally separated from the stack file and lost.

Audio clips can be in WAV, AIFF, or AU format

An audio clip is contained in a stack. Audio clips cannot contain other objects. (An audio clip is not a control, since it has no user interface and cannot be owned by a card.)

The audio clip object has a number of properties and messages associated with it. To see a list of messages that can be sent to an audio clip as a result of user actions or internal Revolution events, open the "Transcript Language Dictionary" page of the main Documentation window, and choose "Audio Clip Messages" from the Show menu at the top. To see a list of all the properties an audio clip can have, choose "Audio Clip Properties" from the Show menu.

autoArm

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

accentColor property, armBorder property, armFill property

Summary

Specifies whether a button becomes armed when the mouse pointer moves into it.

Syntax

set the autoArm of button to {true | false}

Example Code

```
set the autoArm of last button to false
set the autoArm of button "Menu Choices" to true
```

Comments

Use the autoArm property to provide visual feedback to the user about which button the mouse pointer is over. Usually, the autoArm property is used for buttons that are part of stack menus.

Value:

The autoArm of a button is true or false.

By default, the autoArm of newly created buttons is false. Menu items and cascading menu items created with the "New Control" submenu of the Object menu have their autoArm property set to true when created.

Comments:

A button whose autoArm property is set to true does not receive mouseDown messages.

If a button's autoArm property is true, it receives keyDown and keyUp messages for keypresses that occur while the mouse is pressed and the pointer is over the button. If the button's autoArm is false, it does not receive key messages while being clicked.

autoHilite

property

Synonyms
autohilight

Objects
button, field

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
enabled property, hilite property, listBehavior property, style property, How to change the highlight state of a checkbox or button, How to detect the highlight state of a control, How to prevent changing a checkbox's state by clicking it, Why is the selection lost when clicking a button?

Summary
Specifies whether a button highlights when it's pressed, or whether selected text in a field is highlighted.

Syntax
set the autoHilite of {button | field} to {true | false}

Example Code
set the autoHilite of last button to false
if the autoHilite of me is false then beep

Comments
Use the autoHilite property to give visual feedback to users when they click a field or button.

Value:
The autoHilite of a button or field is true or false.

Comments:
If a button's autoHilite property is true, clicking it changes its appearance. This provides visual feedback to the user. (The exact change in appearance depends on the style property of the button and on the current setting of the lookAndFeel property.)

The button remains highlighted while the mouse is within it and the mouse button is down. If the user moves the mouse outside the button while keeping the mouse button depressed, the button becomes unhighlighted. If the user then moves the mouse back over the button, the button becomes highlighted again.

While the button is highlighted, its hilite property is also set to true.

If the button's style property is "checkbox" or "radioButton", clicking it turns the control on or off. If the button's style is menu, the autoHilite has no effect.

If a field's autoHilite property is true, the user can move the text selection by clicking in the field, and text selected with the select command is highlighted. If the field's autoHilite is false, user actions cannot change the selection or place the insertion point in the field by clicking, although a handler can do so with the select command and the user can do so with the arrow keys.

If a field whose listBehavior property is true also has its autoHilite property set to false, list behavior does not work for the field, and a clicked line does not highlight.

autoTab

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

enterKey message, returnInField message, returnKey message, tabKey message, tabStops property

Summary

Specifies whether pressing Return (or Enter) moves to the next field.

Syntax

set the autoTab of field to {true | false}

Example Code

```
set the autoTab of field "First Name" to true
set the autoTab of field myField to false
if the autoTab of the target then go next card
```

Comments

Use the autoTab property to make data entry easier for single-line fields.

Value:

The autoTab of a field is true or false.

By default, the autoTab property of newly created fields is set to false.

Comments:

If the autoTab property of a field is false, pressing the Return key moves the insertion point to the next line of the field.

If the autoTab of a field is true, and the insertion point is already at the last visible line of the field, then pressing Return advances to the next field on the card.

average

function

Synonyms
avg

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

max function, median function, min function, round function, standardDeviation function, sum function

Summary

Returns the arithmetic mean of a list of numbers.

Syntax

average(numbersList)

Example Code

```
average(6,22,8) -- returns 12  
put average(importedList) into field "Imported Price"  
put average(replaceText(field "Values",return,comma)) into avgValue
```

Comments

Use the average function to find the value that best represents a group of values.

Parameters:

The numbersList is a comma-separated list of numbers, or an expression that evaluates to such a list, or an array containing only numbers.

Value:

The average function returns a number.

Comments:

The average of a list of numbers is the sum of the items in the list or elements in the array, divided by the number of items or elements.

The average function can also be written like this:

$\text{sum}(\text{numbersList}) / \text{the number of items in numbersList}$

If the numbersList is empty, the average function returns zero.

Changes to Transcript:

The ability to use an array was introduced in version 1.1. In previous versions, only lists of numbers could be used with the average function.

back

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backScripts function, front keyword, go command, insert script command, remove script command

Summary

Designates a location in the message path after an object's stack, or the most recently visited card.

Syntax

Example Code

```
insert the script of card button 9 into back  
go back
```

Comments

Use the back keyword to return to the most recently visited card, or to designate a backScript.

Comments:

When used with the go command, the back keyword designates the card the user visited most recently. The statement go back returns to that card.

When used with the insert script or remove script command, the back keyword designates a backScript which is to be placed in the message path after the stack.

backdrop

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backgroundColor property, backgroundPattern property, hide command, mouseDownInBackdrop message, mouseUpInBackdrop message, screenRect function, About colors and color references, Color Names Reference, How to hide other applications' windows, Recipe for translating a color name to an RGB numeric triplet, Edit menu > Preferences, View menu > Backdrop

Summary

Places a solid or patterned background behind the application's windows, hiding any other applications.

Syntax

set the backdrop to {colorName | RGBColor | patternNumber | imageID | none}

Example Code

```
set the backdrop to "black"
set the backdrop to "140,30,30"
set the backdrop to none -- removes the backdrop
```

Comments

Use the backdrop property to create a kiosk mode, or to limit distractions during a movie or similar presentation.

Value:

The backdrop is a color reference or a pattern specifier.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

A patternNumber is a built-in pattern number between 1 and 164. (These patterns correspond to Revolution's built-in patterns 136 to 300.)

An imageID is the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the backdrop is "none".

Comments:

Hiding other applications from the user is usually not recommended, since users may need or want to see other windows on their system. However, the ability to do this can be very useful for some applications (such as kiosk systems or games) or at times when you want to reduce distractions (such as during the playing of a movie).

Setting the backdrop property to "none" eliminates the backdrop and lets other windows be seen.

Pattern images can be color or black-and-white.

Cross-platform note: To be used as a pattern on Mac OS systems, an image must be 128x128 pixels or less, and both its height and width must be a power of 2. To be used on Windows and Unix systems, height and width must be divisible by 8. To be used as a fully cross-platform pattern, both an image's dimensions should be one of 8, 16, 32, 64, or 128.

Cross-platform note: On Mac OS systems, if you use the launch or open process commands to start up another application, or if the user brings another application to the front, its windows appear in front of the backdrop. (The backdrop remains visible even if the application is in the background.) Bringing the application to the front again hides the other application behind the backdrop. On Unix and Windows systems, any windows you open after setting the backdrop property remain in front of the backdrop, even if they belong to an application that is in the background.

If the system has multiple screens connected, the backdrop property affects only the main screen.

Pattern images can be color or black-and-white. To be used on Mac OS systems, patterns must be 128x128 pixels or less, and both their height and width must be a power of 2. To be used on Windows and Unix systems, height and width must be divisible by 8.

Changes to Transcript:

The backdrop pattern option was introduced in version 1.1. In previous versions, the backdrop could be a solid color, but not a pattern.

backgroundBehavior

property

Synonyms

bgBehavior

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundIDs property, backgroundNames property, dynamicPaths property, HCAddressing property, pass control structure, send command, About groups and backgrounds, About messages and the message path, How to automatically include groups on a new card

Summary

Specifies whether a group is automatically placed on new cards, and whether the group comes after the cards it's on in the message path.

Syntax

set the backgroundBehavior of group to {true | false}

Example Code

```
set the backgroundBehavior of last group to true
if not the backgroundBehavior of me then send "mouseUp" to me
```

Comments

Use the backgroundBehavior property to make groups behave like HyperCard backgrounds, and to automatically place groups on newly created cards.

Value:

The backgroundBehavior of a group is true or false.

By default, the backgroundBehavior property of newly created groups is set to false.

Note: If a stack created in a version of Revolution earlier than 1.1 is opened in 1.1 or later, the backgroundBehavior of all its groups is set to true by default. This also applies to imported HyperCard stacks.

Comments:

A group's backgroundBehavior property controls two things: the message path of cards containing the group, and whether the group is copied automatically to new cards.

When a message is sent to a card, it is sent next to any groups on the card whose `backgroundBehavior` property is set to `true` (and that haven't already received the message). If you want a group to be in the message path for any card it is placed on, set the group's `backgroundBehavior` to `true`.

When you create a new card, any groups on the current card whose `backgroundBehavior` is `true` are automatically placed on the newly created card. If you create a group of objects that serves as a template for all cards, you should set the group's `backgroundBehavior` to `true` so it will be automatically placed on new cards you create. (Only groups on the current card when you use the create card command are automatically placed on the new card. Other groups are not automatically placed, regardless of their `backgroundBehavior` setting.)

Note: Only groups whose `backgroundBehavior` is `true` appear in the list reported by the `backgroundNames` and `backgroundIDs` properties.

backgroundColor

property

Synonyms

backColor, secondColor, fillBack

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backdrop property, backgroundPattern property, borderColor property, bottomColor property, colorNames function, colors property, effective keyword, focusColor property, foregroundColor property, hiliteColor property, owner property, shadowColor property, topColor property, About colors and color references, Color Names Reference, How to get a striped background in OS X, Why don't buttons respect my color settings?, Recipe for setting the red channel of an object, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies an object's background color.

Syntax

set the backgroundColor of object to {empty | colorName | RGBColor}

set the backgroundColor of [chunk of] field to {empty|colorName|RGBColor}

Example Code

```
set the backgroundColor of last button to "white"
set the backgroundColor of graphic 2 to 128,128,128 -- gray
set the backgroundColor of word thisWord of field "Help!" to "#808080"
get the effective backgroundColor of this card
```

Comments

Use the backgroundColor property to change the background of a window, or the color that fills an object, or to change the background color of text for a highlighted effect.

Value:

The backgroundColor of an object is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the backgroundColor for all objects is empty.

Comments:

Setting the backgroundColor of an object to empty allows the backgroundColor of the object's owner to show through. Use the effective keyword to find out what color is used for the object, even if its own backgroundColor is empty.

If an object's backgroundPattern is set, the pattern is shown instead of the color specified by backgroundColor.

The setting of the backgroundColor property has different effects, depending on the object type:

ï The backgroundColor of a stack or card fills the entire stack window, as well as determining the backgroundColor of each object in the stack or card that does not have its own backgroundColor.

Cross-platform note: On Mac OS, OS X, and Windows systems, if the backgroundColor of all objects in the object hierarchy is empty, the background color set by the system is used.

ï The backgroundColor of a group determines the backgroundColor of each object in the group that does not have its own backgroundColor.

ï The backgroundColor of a button fills the area inside the button's outline. If the button's style is "checkbox", the backgroundColor fills the checkbox. If the button's style is "radioButton", the backgroundColor has no effect. If the button is a tabbed button, the backgroundColor fills the tab area and the frontmost tab, but does not affect the other tabs.

Cross-platform note: If the lookAndFeel is set to "Appearance Manager", standard and rectangle buttons are drawn by the operating system if the backgroundColor and backgroundPattern of the button and all of its owners is empty. (In this case, none of the button's color properties have an effect except for the foregroundColor or foregroundPattern.) Otherwise, the button is drawn by Revolution. If the lookAndFeel is "Appearance Manager", button menus whose menuMode is set to "option" or "comboBox" are always drawn by the operating system, and the setting of the backgroundColor does not affect them.

ï The backgroundColor of a field fills the area inside the field's outline. If you set the backgroundColor of a chunk of a field, only that chunk is affected. For example, to create a "highlighter pen" effect on a single word, set the word's backgroundColor to yellow. If a chunk of text contains runs of text with more than one background color, the backgroundColor of that chunk reports "mixed".

If a field's backgroundColor is empty and the lookAndFeel property is set to "Macintosh", "Appearance Manager", or "Windows 95", the field background is white, instead of inheriting its owner's color.

ï The backgroundColor of a scrollbar fills the arrow boxes at the ends of the scrollbar.

ï The backgroundColor of a graphic is displayed inside the graphic's border if the graphic's fill property is true. If the graphic's fill is false, the setting of the backgroundColor has no effect.

ï The backgroundColor of an EPS object fills the object's rectangle.

ï The backgroundColor of an image is the second color in the image's color palette.

ï The backgroundColor of a player, audio clip, or video clip has no effect.

Changes to Transcript:

The ability of standard buttons to have a backgroundColor under Appearance Manager was introduced in version 2.0. In previous versions, if the lookAndFeel was set to "Appearance Manager", the setting of the backgroundColor had no effect on standard buttons.

The ability to use the system background color was introduced in version 1.1. In previous versions, if the backgroundColor of all objects in the object hierarchy was empty, a light gray color was used.

backgroundIDs

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundBehavior property, backgroundNames property, cardIDs property, groupNames property, ID property, About groups and backgrounds, Why isn't a group listed?

Summary

Reports the backgrounds in a stack.

Syntax

get the backgroundIDs of stack

Example Code

```
get last line of the backgroundIDs of this stack
```

Comments

Use the backgroundIDs property to find out which backgrounds are available.

Value:

The backgroundIDs of a stack reports a list of short ID properties of groups, one per line.

This property is read-only and cannot be set.

Comments:

The backgroundIDs is the list of all backgrounds in the stack whose backgroundBehavior property is set to true, whether they appear on the current card or not. If a group in the stack contains groups, only the top-level groups are reported.

To find out which groups are placed on a card, use the groupIDs property.

Changes to Transcript:

In versions before 1.1, groupIDs and backgroundIDs were synonyms and could be used interchangeably.

backgroundNames

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundBehavior property, backgroundIDs property, cardNames property, groupNames property, About groups and backgrounds, Why isn't a group listed?

Summary

Reports the backgrounds in a stack.

Syntax

get the backgroundNames of stack

Example Code

```
put the backgroundNames of this stack after allObjectsList
```

Comments

Use the backgroundNames property to find out which backgrounds are available.

Value:

The backgroundNames of a stack reports a list of group names, one per line.

This property is read-only and cannot be set.

Comments:

The backgroundNames is the list of all backgrounds in the stack whose backgroundBehavior property is set to true, whether they appear on the current card or not. If a group in the stack contains groups, only the top-level groups are reported.

To find out which groups are placed on a card, use the groupNames property.

Changes to Transcript:

In versions before 1.1, groupNames and backgroundNames were synonyms and could be used interchangeably.

backgroundPattern

property

Synonyms

backPattern, fillPat

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backdrop property, backgroundColor property, borderPattern property, bottomPattern property, brushPattern property, effective keyword, focusPattern property, foregroundPattern property, hilitePattern property, metal property, patterns property, shadowPattern property, topPattern property, How to get a striped background in OS X, Why don't buttons respect my color settings?

Summary

The backgroundPattern specifies the pattern used to draw an object's background.

Syntax

set the backgroundPattern of object to {patternNumber | imageID | empty}

Example Code

```
set the backgroundPattern of this card to 544
set the backgroundPattern of me to the ID of image "Fill"
```

Comments

Use the backgroundPattern property to specify the pattern used for the background on which an image appears, or the pattern used to draw an object.

Value:

The backgroundPattern of an object is a pattern specifier.

A patternNumber is a built-in pattern number between 1 and 164. (These patterns correspond to Revolution's built-in patterns 136 to 300.)

An imageID is the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the backgroundPattern for all objects is empty.

Comments:

Pattern images can be color or black-and-white.

Cross-platform note: To be used as a pattern on Mac OS systems, an image must be 128x128 pixels or less, and both its height and width must be a power of 2. To be used on Windows and Unix systems, height and width must be divisible by 8. To be used as a fully cross-platform pattern, both an image's dimensions should be one of 8, 16, 32, 64, or 128.

The `backgroundPattern` of controls is drawn starting at the control's upper right corner: if the control is moved, the pattern does not shift.

Setting the `backgroundPattern` of an object to empty allows the `backgroundPattern` of the object's owner to show through. Use the `effective` keyword to find out what color is used for the object, even if its own `backgroundPattern` is empty.

The setting of the `backgroundPattern` property has different effects, depending on the object type:

- The `backgroundPattern` of a stack or card fills the entire stack window, as well as determining the `backgroundPattern` of each object in the stack or card that does not have its own `backgroundPattern`.

Cross-platform note: On Mac OS, OS X, and Windows systems, if the `backgroundColor` and `backgroundPattern` of all objects in the object hierarchy is empty, the background set by the system is used.

- The `backgroundPattern` of a group determines the `backgroundPattern` of each object in the group that does not have its own `backgroundPattern`.

- The `backgroundPattern` of a button fills the area inside the button's outline. If the button's style is "checkbox", the `backgroundPattern` fills the checkbox. If the button's style is "radioButton", the `backgroundPattern` has no effect. If the button is a tabbed button, the `backgroundPattern` fills the tab area and the frontmost tab, but does not affect the other tabs.

Cross-platform note: If the `lookAndFeel` is set to "Appearance Manager", standard and rectangle buttons are drawn by the operating system if the `backgroundColor` and `backgroundPattern` of the button and all of its owners is empty. Otherwise, the button is drawn by Revolution. If the `lookAndFeel` is "Appearance Manager", button menus whose `menuMode` is set to "option" or "comboBox" are always drawn by the operating system, and the setting of the `backgroundPattern` does not affect them.

- The `backgroundPattern` of a field fills the area inside the field's outline and (if the field is a scrolling field) the arrow boxes at the ends of the scrollbar. If you set the `backgroundPattern` of a chunk of a field (on Unix systems), only that chunk is affected.

- The `backgroundPattern` of a scrollbar fills the arrow boxes at the ends of the scrollbar.

- The `backgroundPattern` of a graphic is displayed inside the graphic's border if the graphic's `fill` property is true. If the graphic's `fill` is false, the setting of the `backgroundPattern` has no effect.

- The `backgroundPattern` of an EPS object fills the object's rectangle.

- The `backgroundPattern` of an image, audio clip, video clip, or player has no effect.

If an object's `backgroundPattern` is set, the pattern is shown instead of the color specified by `backgroundColor`.

Changes to Transcript:

The ability to use the system background pattern was introduced in version 1.1. In previous versions, if the `backgroundColor` and `backgroundPattern` of all objects in the object hierarchy was empty, a light gray color was used.

backgroundPixel

property

Synonyms

backPixel, secondPixel

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundColor property, borderPixel property, bottomPixel property, colorMap property, focusPixel property, foregroundPixel property, hilitePixel property, screenColors function, shadowPixel property, topPixel property, Recipe for setting the red channel of an object, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies which entry in the color table is used for an object's background color.

Syntax

set the backgroundPixel of object to colorNumber

Example Code

```
set the backgroundPixel of button 3 to field "Selected Color"
```

Comments

Use the backgroundPixel property to change the background color of an object when the bit depth of the screen is 8 bits (256 colors) or less.

Value:

The backgroundPixel of an object is an integer between zero and (the screenColors - 1). It designates an entry in the current color table.

By default, the backgroundPixel for all objects is empty.

Comments:

The backgroundPixel property specifies which entry in the color table is used for an object's background color. It is similar to the backgroundColor property, but is specified as an entry in the current color table, rather than as a color reference.

The color table can be set by changing the colorMap property.

backScripts

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

back keyword, frontScripts function, ID property, insert script command, remove script command, scriptLimits function, stacksInUse property

Summary

Returns a list of objects that have been inserted into the message path after the stack the target object belongs to.

Syntax

the backScripts

backScripts()

Example Code

```
the backScripts
if the long ID of me is not among the lines of the backScripts then beep
```

Comments

Use the backScripts function to find out which scripts receive messages and function calls after the current object, the objects that own the current object, and any stacks in the stacksInUse.

Value:

The backScripts function returns a list of the long ID property of all objects that have been inserted into the back, one ID per line.

Comments:

A script inserted into the back with the insert script command receives messages after all objects in the message path, just before the application itself receives the message.

If more than one object is in the backScripts, their order in the message path is the same as their order in the list. For example, the first object in the backScripts receives messages before the second object. This order is the reverse of the order in which the objects were added with the insert script command.

When using the development environment, you can place as many objects as you want in the backScripts. When using a standalone application, the number of backScripts that can be active at once is specified by the scriptLimits function, and is currently ten.

backSize

property

Synonyms

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

formattedRect property, height property, rectangle property, width property

Summary

Specifies the width and height of the specified background.

Syntax

set the backSize of group to width,height

Example Code

```
set the backSize of group "Nav Bar" to 160,20
```

Comments

Use the backSize property to hide members of a group from view.

Value:

The backSize of a group consists of two positive integers separated by a comma.

Comments:

The first item of the backSize is the width of the group in pixels. The second item is the height of the group in pixels.

If you change the backSize of a group, the position of its top left corner remains unchanged.

This property is included in Transcript for compatibility with imported SuperCard projects.

backslash

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

slash constant, constant command

Summary

Equivalent to the character (ASCII 92).

Syntax

Example Code

```
put theDrive & backslash & theNode into windowsStylePath
```

Comments

Use the backslash constant as an easier-to-read replacement for ".".

backspaceKey

message

Synonyms

Objects

button, field, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cutKey message, delete command, deleteKey message

Summary

Sent to the active (focused) control, or to the current card if no control is focused.

Syntax

backspaceKey

Example Code

```
on backspaceKey -- in the script of a field
  -- clear the whole field instead of removing one character:
  put empty into me
end backspaceKey
```

Comments

Handle the backspaceKey message if you want to do something special when the user presses the Backspace key.

Comments:

The Backspace key is not the same as the Forward Delete key. On some keyboards, the Backspace key is labeled "Delete".

base64Decode

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

base64Encode function, binaryDecode function, charToNum function, decompress function, numToChar function, URLDecode function

Summary

Returns the original data from a base 64-encoded string.

Syntax

the base64Decode of encodedData

base64Decode(encodedData)

Example Code

```
put base64Decode(receivedData) into URL "file:download.mov"  
baseDecode(baseEncode(myData)) -- returns myData
```

Comments

Use the base64Decode function to take a base 64-encoded string and transform it back into the original binary data.

Parameters:

The encodedData is a string of of any length. Valid base 64-encoded data can include uppercase and lowercase letters, digits, +, /, and =. Other characters are ignored.

Value:

The base64Decode function returns the original binary data.

Comments:

The base64Decode function is the inverse of the base64Encode function.

The decoded result is generally smaller than the encoded data.

Base 64-encoded data does not necessarily contain any = signs. If a run of one or more = signs is present, it indicates the end of the data.

The base 64 encoding scheme is often used to encode binary data for MIME mail and HTTP transfers.

For technical information about base 64 encoding, see section 6.8 of RFC 2045 at [<http://www.ietf.org/rfc/rfc2045.txt>](http://www.ietf.org/rfc/rfc2045.txt).

base64Encode

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

base64Decode function, binaryDecode function, charToNum function, compress function, numToChar function, URLDecode function

Summary

Returns a base 64-encoded string.

Syntax

the base64Encode of data

base64Encode(data)

Example Code

```
write base64Encode(pictureFileData) to socket thisSocket
```

Comments

Use the base64Encode function to encode binary data for transmission over communication channels that don't accept raw binary data.

Parameters:

The data is a string of binary data of any length.

Value:

The base64Encode function returns a text string representing the binary data.

Comments:

The base64Encode function is the inverse of the base64Decode function. The encoded result is generally about a third larger than the original data.

The encoded string returned by base64Encode function can include uppercase and lowercase letters, digits, +, /, and =, but no other characters. Base 64-encoded data is wrapped at 72 characters, so each line of the encoded data is 72 characters or (for the last or only line) fewer.

The base 64 encoding scheme is often used to encode binary data for MIME mail and HTTP transfers.

For technical information about base 64 encoding, see section 6.8 of RFC 2045 at <http://www.ietf.org/rfc/rfc2045.txt>.

baseConvert

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

convertOctals property

Summary

Converts a number from one base to another.

Syntax

baseConvert(number,originalBase,destinationBase)

Example Code

```
baseConvert(16,10,2) -- yields 10000, which is 16 in base 2
baseConvert(27,10,16) -- yields 1B, which is 27 in base 16
baseConvert("1C",16,10) -- yields 28, the base-10 equivalent of 1C
```

Comments

Use the baseConvert function to provide input or output of numbers in a base other than base 10: for example, in hexadecimal (base 16) or binary (base 2).

Parameters:

The number is the number to be converted, expressed in its original base. The number must be an integer between zero and 4,294,967,295 ($2^{32} - 1$). If the number includes non-digits (as, for example, a base-16 number can include letters A-F), enclose the number in quotes.

The originalBase is an integer between 2 and 36.

The destinationBase is an integer between 2 and 36.

Value:

The baseConvert function returns a number, expressed in the destinationBase.

Comments:

The everyday decimal number system is called "base 10" because we count from 1 to 9, and the tenth digit moves over to the tens place and is written 10: one group of ten, plus zero extra ones. Similarly, a number like 384 means one group of a hundred (10^2), plus eight groups of ten, plus four leftover ones.

It is possible to write numbers in other bases. For example, suppose you want to write the number six in base 4. In base 4, we count from 1 to 3, and the fourth digit moves over to the "fours place". So the numbers from one to six, in base 4, are written "1, 2, 3, 10, 11, 12". The number 12 in base 4 means one group of four, plus two leftover ones. This same number is written as 6 in base 10.

If the base is greater than 10, then digits greater than 9 are expressed as capital letters: A is the digit ten, B is the digit eleven, and so on.

Transcript always does math in base 10, so if you want to perform mathematical calculations such as addition on a number in another base, you must first convert the number to base 10, do the calculation, then convert back to the original base. Here is an example:

```
function hexSum firstAddend, secondAddend
  -- adds together two hexadecimal numbers
  put baseConvert(firstAddend,16,10) into convertedAddend1
  put baseConvert(secondAddend,16,10) into convertedAddend2
  put convertedAddend1 + convertedAddend2 into baseTenSum
  return baseConvert(baseTenSum,10,16)
end hexSum
```

beep

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

beepDuration property, beepLoudness property, beepPitch property, play command, sound function, How to beep, How to make the computer speak out loud

Summary

Sounds the system beep.

Syntax

beep [numberOfTimes]

Example Code

```
beep
beep 7
beep trunc(field "Severity Level")
```

Comments

Use the beep command to get the user's attention.

Parameters:

The numberOfTimes is an integer (or an expression that evaluates to an integer) and specifies how many beeps are sounded. If the numberOfTimes is not specified, one beep is sounded.

Comments:

If the numberOfTimes is less than 1, the beep command has no effect (but does not cause a script error).

Cross-platform note: Windows and OS X do not execute the beep command if it's issued while a beep is playing. This means that if you specify a numberOfTimes on a Windows or OS X system, the user might hear fewer beeps because not all of them are sent to the speaker. To ensure that the user hears a specific number of beeps, use a loop with a wait command (where the wait time is at least as long as the beep sound's duration) after each beep:

```
repeat for 4 times -- ensure 4 separate beeps
  beep
```



```
wait 200 milliseconds  
end repeat
```

Note: Overuse of the beep command has been known to cause user stress.

beepDuration

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

beep command, beepLoudness property, beepPitch property, play command, sound function

Summary

Sets the length of the sound played by the beep command.

Syntax

set the beepDuration to milliseconds

Example Code

```
set the beepDuration to 2000 -- two seconds
set the beepDuration to myBeep * 1000
```

Comments

Use the beepDuration property to vary the length of the beep, in order to increase or decrease its urgency.

Value:

The beepDuration is an integer between -1 and 65535.

By default, the beepDuration is 500 (half a second).

Comments:

Set the beepDuration to -1 to use the default system setting.

The beepDuration has no effect on Mac OS or Windows systems, and on some Unix systems which don't allow changing the beep length.

beepLoudness

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

beep command, beepDuration property, beepPitch property, playLoudness property, sound function

Summary

Sets the volume of the sound played by the beep command.

Syntax

set the beepLoudness to level

Example Code

```
set the beepLoudness to 0
set the beepLoudness to 50
put the beepLoudness into storedBeepVolume
```

Comments

Use the beepLoudness property to vary the volume of the beep, in order to increase or decrease its urgency.

Value:

The beepLoudness is an integer between zero and 100.

Comments:

Set the beepLoudness to -1 to use the default system setting.

The beepLoudness has no effect on Mac OS or Windows systems, and on some Unix systems which don't allow changing the beep loudness.

beepPitch

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

beep command, beepDuration property, beepLoudness property, play command, revSetSpeechPitch command, sound function

Summary

Sets the frequency of the sound played by the beep command.

Syntax

set the beepPitch to pitchFrequency

Example Code

```
set the beepPitch to 220 -- A below middle C
set the beepPitch to -1 -- restores default
```

Comments

Use the beepPitch property to make the beep sound higher or lower.

Value:

The beepPitch is an integer between zero and 20000.

Comments:

The pitchFrequency is the frequency in Hz (cycles per second). A value of 440 is A above middle C. Halving the frequency lowers the pitch one octave; doubling the frequency raises it one octave.

Set the beepPitch to -1 to use the default system setting.

The beepPitch has no effect and does not report a meaningful value on Mac OS or Windows systems, and on some Unix systems which don't allow changing the beep pitch.

before
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

after keyword, into keyword, put command, select command, Recipe for adding a prefix to each line of a string

Summary

Used with the put command to place a value at the start of a container or chunk.

Syntax

Example Code

```
put return before myVariable  
put thisValue before line 17 of field "Results"
```

Comments

Use the before keyword to position the insertion point at a selected location in a field, or to add text at a specific location in a container.

Comments:

You can specify either a container, or a chunk within a container. If you don't specify a chunk, the before keyword specifies the start of the container.

When you use the before keyword, the current contents of the container or chunk is not affected. The text is added to the container or chunk, instead of replacing it.

binaryDecode

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

base64Decode function, binaryEncode function, decompress function, numToChar function

Summary

Decodes binary data and places it into the specified variables.

Syntax

`binaryDecode(formatsList,data,variablesList)`

Example Code

```
binaryDecode("h","M",theHex) - converts M to its hex equivalent  
binaryDecode("a*",receivedData,textData) -- converts data to text  
binaryDecode("x12Sh16",picHeader,junk,numColors,colorTable)  
binaryDecode(myFormat,placeholder,importantStuff)
```

Comments

Use the binaryDecode function to convert binary data into a form that can be manipulated by handlers.

Parameters:

The formatsList consists of one or more dataTypes, each followed optionally by an amount. A dataType is one of the following single letters:

- x: skip next amount bytes of data
- a: convert next amount bytes of data to characters
- A: convert next amount bytes of data (skipping spaces) to characters
- b: convert next amount bits of data to 1s and 0s
- B: convert next amount bits of data to 1s and 0s, starting at the high end of each byte
- h: convert next amount bytes of data to hexadecimal digits
- H: convert next amount bytes of data to hexadecimal digits, starting at the high end of each byte
- c: convert next amount bytes of data to signed 1-byte integers
- C: convert next amount bytes of data to unsigned 1-byte integers
- s: convert next amount 2-byte chunks of data to signed integers in host byte order
- S: convert next amount 2-byte chunks of data to unsigned integers in host byte order
- i: convert next amount 4-byte chunks of data to signed integers in host byte order

I: convert next amount 4-byte chunks of data to unsigned integers in host byte order
n: convert next amount 2-byte chunks of data to signed integers in network byte order
N: convert next amount 4-byte chunks of data to unsigned integers in network byte order
f: convert next amount 4-byte chunks of data to a single-precision float
d: convert next amount 8-byte chunks of data to a double-precision float

The amount corresponding to each dataType is an integer or the * character. If no amount is specified, the dataType is used for a single byte of data. The * character causes the rest of the data to be converted according to the formatType, so it should appear only as the last character in the formatsList.

Important! If you specify an amount with a string dataType (a or A), the binaryDecode function places amount bytes in the next variable of the variablesList. If you specify an amount with a numeric dataType, the function places amount chunks of data in the next amount variables of the variablesList. For example, the dataType "a3" requires only one variable, which will hold a 3-character string, but the dataType "h3" requires three variables, each of which will hold a single hex digit.

The data is a string of encoded binary data.

The variablesList is a comma-separated list of local or global variable names. The number of variable names must be the same as the number of dataTypes specified in the formatsList, and the variables must already exist.

Value:

The binaryDecode function returns the number of dataTypes that were successfully converted and placed in variables (not counting data skipped by the x dataType).

The actual data is placed in the variables, rather than returned by the function.

Comments:

The binary data format used by binaryDecode is similar to the format produced by the "pack" function of the Perl programming language.

You must declare or otherwise create all variables in the variablesList before using them in the binaryDecode function. Unlike the put command, the binaryDecode function does not automatically create local variables when you use them.

Although the x dataType skips the specified number of bytes rather than converting them, you still must provide a variable for instances of x that appear in the formatsList. The binaryDecode function does not change the value of a variable used for the dataType x.

If the formatsList specifies fewer bytes than are in the data, the binaryDecode function ignores the remaining bytes. If the formatsList specifies more bytes than are in the data, the binaryDecode function converts as many of the dataTypes as there is data for. Check the value that the binaryDecode function returns to determine how much data was actually converted. Here is an example:

```
on convertStuff dataToConvert
  global headerData,placeholder,bodyData,footerData
  put binaryDecode("i5x2a24xi2",dataToConvert,
```

```
headerData,placeholder,bodyData,placeholder,footerData)  
  
    into convertResult  
    if convertResult < 3 then return "Error: data was corrupted"  
end convertStuff
```


binaryEncode

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

base64Encode function, binaryDecode function, charToNum function, compress function, null constant

Summary

Encodes a set of values into a single binary value.

Syntax

binaryEncode(formatsList,dataStringList)

Example Code

```
binaryEncode("a","Q") -- returns "Q"  
binaryEncode("A5","QED") -- returns "QED  "  
binaryEncode("N2a12x",firstNum,secondNum,labelString,"nothing")  
charToNum(binaryEncode("B*", "01111111")) -- returns 127
```

Comments

Use the binaryEncode function to encode one or more values as binary data.

Parameters:

The formatsList consists of one or more dataTypes, each followed optionally by an amount. A dataType is one of the following single letters:

x: output amount null characters
a: encode amount characters as characters, padding with nulls
A: encode amount characters as characters, padding with spaces
b: encode amount groups of 4 1s and 0s as bits
B: encode amount groups of 4 1s and 0s as bits, starting at the high end of each byte
h: encode amount groups of 2 characters as hexadecimal numbers
H: encode amount groups of 2 characters as hexadecimal, starting at the high end of each byte
c: encode amount numbers as signed 1-byte integers
C: encode amount numbers as unsigned 1-byte integers
s: encode amount numbers as signed 2-byte integers in host byte order
S: encode amount numbers as unsigned 2-byte integers in host byte order
i: encode amount numbers as signed 4-byte integers in host byte order

I: encode amount numbers as unsigned 4-byte integers in host byte order
n: encode amount numbers as signed 2-byte integers in network byte order
N: encode amount numbers as signed 4-byte integers in network byte order
f: encode amount numbers as single-precision floating-point numbers
d: encode amount numbers as double-precision floating-point numbers

The amount corresponding to each dataType is an integer or the * character:

ĩ If the dataType is a, A, b, B, h, or H, the amount specifies the number of characters or groups of the dataString to use; extra characters are ignored. The * character encodes the rest of the data in the current dataString. If no amount is specified, the dataType is used for one character.

ĩ If the dataType is c, C, s, S, i, I, n, N, f, or d, the amount specifies the number of dataStrings to encode. The * character encodes the rest of the dataStrings. If no amount is specified, the dataType is used for one dataString.

ĩ If the dataType is x, the amount specifies how many nulls to place in the returned value.

The dataStringList is a comma-separated list of dataStrings. Each dataString is a string, or an expression that evaluates to a string.

Value:

The binaryEncode function returns the binary string representation of the dataStrings.

Comments:

The binary data format produced by binaryEncode is similar to the format produced by the "pack()" function of the Perl programming language.

Although the x dataType places nulls in the resulting string regardless of the contents of its corresponding dataString, you still must provide a dataString for instances of x that appear in the formatsList. The binaryEncode function disregards the contents of the dataString used for the dataType x, so you can use any value.

If you don't specify a number with the a, A, b, B, h, or H dataTypes, one character is encoded. If the dataType normally takes more than one character, trailing zeroes are added to make up the required number of characters. For example, H requires two characters, so binaryEncode("H","3") encodes the hexadecimal number 30, adding a zero to the end to make the dataString two characters long.

binfile

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

file keyword, http keyword, resfile keyword, URL keyword, About filename specifications and file paths, About using URLs, uploading, and downloading, How to associate files with a Mac OS standalone application, How to associate files with an OS X standalone application, How to associate files with a Windows standalone application, How to create a file, Why is there a problem with line endings?

Summary

Used as a URL type with the get and put commands to designate a local file.

Syntax

Example Code

```
answer myConversionFunction(URL "binfile:input")  
put imageData into URL "binfile:/Main/project/test.gif"
```

Comments

Use the binfile keyword to work with binary files.

Comments:

The binfile scheme indicates a binary file which is located on the user's system. The file is specified by either a full path starting with "/", or a relative path starting from the defaultFolder.

A URL container can be used anywhere another container type is used.

When you put data into a binfile URL or get data from it, Revolution does not translate end-of-line markers (ASCII 10 and ASCII 13) between the current platform's standard and Revolution's internal standard of a linefeed. This ensures that binary data, which may contain such characters, is not accidentally corrupted.

If you are working with text data (such as text in fields), use the file URL scheme instead.

bitAnd operator

Synonyms

Objects numeric

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

and operator, baseConvert function, bitNot operator, bitOr operator, bitXOr operator, srcAnd keyword, Operator Precedence Reference

Summary

Performs a "bitwise and" operation on the binary representation of two numbers.

Syntax

number1 bitAnd number2

Example Code

```
0 bitAnd 1 -- evaluates to 0  
3 bitAnd 2 -- in binary: 11 bitAnd 10; evaluates to 10; converted to 2
```

Comments

Use the bitAnd operator to operate directly on the bits of two numbers.

Parameters:

The number1 and number2 are numbers, or expressions that evaluate to numbers, between zero and 4,294,967,295 ($2^{32} - 1$).

Comments:

To perform the bitAnd operation, Revolution first converts both operands to their binary equivalent, a string of ones and zeroes. 1 is equivalent to true, and 0 is equivalent to false.

For each bit of number1, Revolution performs an and operation with the corresponding bit of number2 to produce a result. A bit is 1 if the corresponding bits of number1 and number2 are both 1. Otherwise, the bit is 0.

Finally, the binary number thus created is converted back to decimal.

bitNot

operator

Synonyms

Objects

numeric

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

baseConvert function, bitAnd operator, bitOr operator, bitXOr operator, not operator, Operator Precedence Reference

Summary

Performs a "bitwise not" operation on the binary representation of a number.

Syntax

bitNot number

Example Code

```
bitNot 1 -- evaluates to 0  
bitNot 14 -- in binary: bitNot 1110; evaluates to 0001; converted to 1
```

Comments

Use the bitNot operator to operate directly on the bits of a number.

Parameters:

The number is a number, or an expression that evaluates to a number, between zero and 4,294,967,295 ($2^{32} - 1$).

Comments:

To perform the bitNot operation, Revolution first converts the operand to its binary equivalent, a string of ones and zeroes. 1 is equivalent to true, and 0 is equivalent to false.

For each bit of the number, Revolution performs a not operation. A bit is 0 if the corresponding bit of the number is 1, and 1 if the corresponding bit of the number is 0.

Finally, the binary number thus created is converted back to decimal.

bitOr

operator

Synonyms

Objects

numeric

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

baseConvert function, bitAnd operator, bitNot operator, bitXOr operator, or operator, srcOr keyword, Operator Precedence Reference

Summary

Performs a "bitwise or" operation on the binary representation of two numbers.

Syntax

number1 bitOr number2

Example Code

```
1 bitOr 0 -- evaluates to 1
6 bitOr 2 -- in binary: 110 bitOr 010; evaluates to 110; converted to 6
```

Comments

Use the bitOr operator to operate directly on the bits of two numbers.

Parameters:

The number1 and number2 are numbers, or expressions that evaluate to numbers, between zero and 4,294,967,295 ($2^{32} - 1$).

Comments:

To perform the bitOr operation, Revolution first converts both operands to their binary equivalent, a string of ones and zeroes. 1 is equivalent to true, and 0 is equivalent to false.

For each bit of number1, Revolution performs an or operation with the corresponding bit of number2 to produce a result. A bit is 0 if the corresponding bits of number1 and number2 are both 0. Otherwise, the bit is 1.

Finally, the binary number thus created is converted back to decimal.

bitXor

operator

Synonyms

Objects

numeric

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

bitAnd operator, bitNot operator, bitOr operator, or operator, srcXOr keyword, Operator Precedence Reference

Summary

Performs a "bitwise exclusive or" on the binary representation of two numbers.

Syntax

number1 bitXOr number2

Example Code

```
1 bitXOr 0 -- evaluates to 1
6 bitXOr 4 -- in binary: 110 bitXOr 100; evaluates to 010; converts to 2
```

Comments

Use the bitXOr operator to operate directly on the bits of two numbers.

Parameters:

The number1 and number2 are numbers, or expressions that evaluate to numbers.

Comments:

To perform the bitXOr operation, Revolution first converts both operands to their binary equivalent, a string of ones and zeroes. 1 is equivalent to true, and 0 is equivalent to false.

For each bit of number1, Revolution performs an exclusive or operation with the corresponding bit of number2 to produce a result. A bit is 0 if the corresponding bits of number1 and number2 are both 0 or both 1. Otherwise, the bit is 1.

Finally, the binary number thus created is converted back to decimal.

black

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

card keyword, gray keyword, hide command, inverse keyword, show command, visual effect command, white keyword

Summary

Used with the visual effect command to indicate that a blank black screen is shown at the end of the visual effect.

Syntax

Example Code

```
visual effect push left to black
```

Comments

Use the black keyword to transition to a blank black background in a sequence of visual effects.

Comments:

Visual effects can be stacked in a sequence by using several visual effect commands in succession. If the last transition ends with showing the card, and all except the last one shows an intermediate image (such as a solid black color), the effect is enhanced. You show a solid black color at the end of a transition by using the black keyword.

This example uses the push effect with the black keyword to emulate an old-fashioned slide projector:

```
visual effect push left to black -- from card to solid black  
visual effect push right to card -- from black to final card  
go card "Destination"
```


blend

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blendLevel property, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of button "My Button" to blend
```

Comments

Use the blend keyword to combine an object's color with the colors underneath it.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed. When the blend mode is used, each component of the object color (red, green, and blue) is compared with the corresponding component of the color underneath, and the average of each component is used for the displayed color.

For example, suppose an object's color is 30,70,150, and the color of the pixels under the object is 60,40,100. If the blend mode is used, the object's displayed color is 45,55,125.

Whenever you set the blendLevel property of an image, its ink property is automatically set to "blend".

Important! The setting of the blend property has no effect on images in PICT format.

blendLevel

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

alphaData property, blend keyword, ink property, opaque property, Shortcut to make an image transparent, Tip: Fadeout effect using blendLevel

Summary

Specifies the degree of transparency of an image whose ink property is set to "blend".

Syntax

set the blendLevel of image to levelNumber

Example Code

```
set the blendLevel of image 3 to 100
```

```
set the blendLevel of image "Test" to the blendLevel of image "Test" - 1
```

Comments

Use the blendLevel property to allow objects underneath an image to show through partially or completely.

Value:

The blendLevel of an image is an integer between zero and 100.

By default, the blendLevel property of newly created images is set to 50.

Comments:

If an image's blendLevel is zero, the image is fully opaque. If the blendLevel is 100, the image is fully transparent. Values between zero and 100 indicate levels of partial translucency.

Whenever you set the blendLevel property of an image, its ink property is automatically set to "blend". If the image's ink property is not set to "blend", the setting of its blendLevel has no effect.

Tip: The blendLevel property specifies the transparency of the entire image at once. To set transparency of part of an image, pixel-by-pixel, see the alphaData property.

blindTyping

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

message box keyword, type command, visible property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the blindTyping to {true | false}

Example Code

Comments

In HyperCard, the blindTyping property determines whether the user can type into the message box when the message box is not visible.

In Revolution, the message box always acts as though the blindTyping is set to false: it must be visible for the user to type into it.

blinkRate

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

idleRate property

Summary

Specifies the interval in milliseconds between the time the insertion point blinks on and the time it blinks off.

Syntax

set the blinkRate to blinkDelayInMilliseconds

Example Code

```
set the blinkRate to 1000
```

```
if the blinkRate > 400 then set the borderColor to red
```

Comments

Use the blinkRate property to control how fast the text insertion point blinks.

Value:

The blinkRate is an integer between 1 and 65535.

By default, the blinkRate is 200 (one-fifth of a second).

Comments:

The blinkRate should be a multiple of the idleRate.

bold

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

italic keyword, plain keyword, textStyle property, underline keyword, Text menu > Bold

Summary

Used with the textStyle property to indicate boldface text.

Syntax

Example Code

```
set the textStyle of button "Label" to bold
if the textStyle of word 3 of field "Info" is bold then go card 1
```

Comments

Use the bold keyword to emphasize text by presenting it in boldface.

Comments:

You can boldface an object (which boldfaces all the text displayed by the object) or a chunk in a field or button.

To add boldfacing to an object without removing other styles (such as italic), add the bold keyword to the end of the textStyle. The following example adds bold to the styles of a button:

```
if the textStyle of button 1 is empty then
  set the textStyle of button 1 to bold
else
  set the textStyle of button 1 to
```

```
(the textStyle of button 1) & comma & "bold"
end if
```

borderColor

property

Synonyms

fourthColor, markerFillColor

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundColor property, borderPattern property, borderWidth property, bottomColor property, colorNames function, colors property, effective keyword, focusColor property, foregroundColor property, hiliteColor property, owner property, shadowColor property, showBorder property, threeD property, topColor property, About colors and color references, Color Names Reference, Why don't buttons respect my color settings?, Recipe for translating a color name to an RGB numeric

Summary

Specifies the color of an object's border.

Syntax

set the borderColor of object to {empty | colorName | RGBColor}

Example Code

```
set the borderColor of field "Alert" to "red"
set the borderColor of last button to "#000000"
set the borderColor of scrollbar ID 1010 to 128,128,128
```

Comments

Use the borderColor property to specify the border color of a button or scrollbar.

Value:

The borderColor of an object is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the borderColor for all objects is empty.

Comments:

Setting the `borderColor` of an object to empty allows the `borderColor` of the object's owner to show through. Use the `effective` keyword to find out what color is used for the object, even if its own `borderColor` is empty.

If the object's `showBorder` property is false, the `borderColor` has no effect. If the object's `threeD` property is true, the `topColor` and `bottomColor` are used to draw the border, rather than the `borderColor`.

The setting of the `borderColor` property has different effects, depending on the object type:

- ī The `borderColor` of a stack determines the `borderColor` of each object in the stack that does not have its own `borderColor`.

- ī The `borderColor` of a card or group determines the color of the border around the card or group, as well as determining the `borderColor` of each object in the card or group that does not have its own `borderColor`.

- ī The `borderColor` of a button determines the color of the border around the button. If the button's style is "checkbox" or "radioButton", the `borderColor` has no effect. If the button is a button menu, the `borderColor` has no effect unless the button's `menuMode` property is set to "comboBox" or "popup". If the button's `threeD` property is set to true, the `borderColor` has no effect regardless of the button's style.

- ī The `borderColor` of a field determines the color of the border around a scrolling field's scrollbar, the color of the grid lines if the field's `hGrid` or `vGrid` property is true, and the color of the outline around any text in the field whose `textStyle` is set to "box".

- ī The `borderColor` of a scrollbar is the color of the border surrounding the scrollbar.

- ī The `borderColor` of a graphic is used to fill the marker shapes at the graphic's vertexes, if the graphic's style is curve or polygon and its `markerDrawn` is true. It is also used to draw a border around it if the graphic's `showBorder` and `threeD` properties are both set to true.

- ī The `borderColor` of an audio clip or video clip has no effect.

- ī The `borderColor` of a player or EPS object is the color of the border around the object. If the object's `threeD` property is set to true, the `borderColor` has no effect.

Tip: To set the color of an image's border, set the `borderColor` of the card, stack, or group that owns the image.

If an object's `borderPattern` is set, the pattern is shown instead of the color specified by `borderColor`.

Changes to Transcript:

The `borderColor`'s effect on grid lines in fields was introduced in version 2.0. In previous versions, the color of the grid lines was determined by the field's `hiliteColor` property.

borderPattern

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

borderColor property, borderWidth property, effective keyword, owner property, patterns property, threeD property

Summary

Specifies the pattern of an object's border.

Syntax

set the borderPattern of object to {empty | patternID | imageID}

Example Code

```
set the borderPattern of group "Nav" to myPattern
```

Comments

Use the borderPattern property to specify the border pattern of a button or scrollbar.

Value:

The borderPattern of an object is a pattern specifier.

A patternNumber is a built-in pattern number between 1 and 164. (These patterns correspond to Revolution's built-in patterns 136 to 300.)

An imageID is the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the borderPattern for all objects is empty.

Comments:

Pattern images can be color or black-and-white.

Cross-platform note: To be used as a pattern on Mac OS systems, an image must be 128x128 pixels or less, and both its height and width must be a power of 2. To be used on Windows and Unix systems,

height and width must be divisible by 8. To be used as a fully cross-platform pattern, both an image's dimensions should be one of 8, 16, 32, 64, or 128.

The borderPattern of controls is drawn starting at the control's upper right corner: if the control is moved, the pattern does not shift.

Setting the borderPattern of an object to empty allows the borderPattern of the object's owner to show through. Use the effective keyword to find out what color is used for the object, even if its own borderPattern is empty.

If the object's showBorder property is false, the borderPattern has no effect.

The setting of the borderPattern property has different effects, depending on the object type:

- ï The borderPattern of a stack, card, or group determines the borderPattern of each object in the stack, card, or group that does not have its own borderPattern.
- ï The borderPattern of a button determines the pattern of the border around the button. If the button's style is checkbox or radioButton, the borderPattern has no effect. If the button is a button menu, the borderPattern has no effect unless the button's menuMode property is set to comboBox or popup. If the button's threeD property is set to true, the borderPattern has no effect regardless of the button's style.
- ï The borderPattern of a field has no effect, except for the border around a scrolling field's scrollbar. The field's borderPattern also determines the pattern of the outline around any text in the field whose textStyle is set to "box".
- ï The borderPattern of a scrollbar determines the pattern of the border surrounding the scrollbar.
- ï The borderPattern of a graphic, audio clip, video clip, or image has no effect.
- ï The borderPattern of a player or EPS object determines the pattern of the border around the object. If the object's threeD property is set to true, the borderPattern has no effect.

If an object's borderPattern is set, the pattern is shown instead of the color specified by borderColor.

borderPixel

property

Synonyms

fourthPixel

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundPixel property, borderColor property, bottomPixel property, colorMap property, focusPixel property, foregroundPixel property, hilitePixel property, screenColors function, shadowPixel property, topPixel property, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies which entry in the color table is used for the color of an object's border.

Syntax

set the fourthPixel of object to colorNumber

Example Code

```
set the fourthPixel of group 1 to 200
```

Comments

Use the borderPixel property to change the border color of an object when the bit depth of the screen is 8 bits (256 colors) or less.

Value:

The borderPixel of an object is an integer between zero and (the screenColors - 1). It designates an entry in the current color table.

By default, the borderPixel for all objects is empty.

Comments:

The borderPixel property specifies which entry in the color table is used for an object's border color. It is similar to the borderColor property, but is specified as an entry in the current color table, rather than as a color reference.

The color table can be set by changing the colorMap property.

borderWidth

property

Synonyms

lineSize

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

borderColor property, borderPattern property, margins property, shadowOffset property, showBorder property, threeD property

Summary

Specifies the width of an object's border.

Syntax

set the borderWidth of object to pixelWidth

Example Code

```
set the borderWidth of button "Cancel" to 1  
set the borderWidth of this card to 24
```

Comments

Use the borderWidth property to change the appearance of an object's border.

Value:

The borderWidth of an object is an integer between zero and 255.

By default, the borderWidth of newly created objects is 2.

Comments:

If the object's showBorder property is false, the borderWidth property has no effect.

The border's appearance and color is affected by the setting of the object's threeD property.

bottom

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomLeft property, bottomMargin property, bottomRight property, height property, rectangle property, top property, Object menu > Align Selected Controls > Bottom, Shortcut to align control edges

Summary

Specifies how far an object's bottom edge is from the top of the window or screen.

Syntax

set the bottom of object to number

Example Code

```
if the bottom of this stack > 600 then changeStackPlacement  
set the bottom of card field 1 to the height of this stack
```

Comments

Use the bottom property to change the vertical placement of a control or window.

Value:

The bottom of an object is an integer. A negative integer indicates that the position is above the top of the screen or card (and therefore cannot be seen).

A stack's bottom is the distance in pixels from the top edge of the screen.

A card's or control's bottom is the distance in pixels from the top edge of the card to the bottom edge of the card or control.

Comments:

The bottom of a stack is in absolute (screen) coordinates. The bottom of a card is always equal to the height of the stack window; setting the bottom of a card does not cause a script error, but it has no effect. The bottom of a group or control is in relative (window) coordinates.

Changing the bottom of an object shifts it to the new position without resizing it. To change an object's height, set its height or rectangle property.

The height property of an object is equal to its bottom minus its top.

bottomColor

property

Synonyms

sixthColor

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundColor property, borderColor property, bottomPattern property, colorNames function, colors property, effective keyword, focusColor property, foregroundColor property, hiliteColor property, owner property, shadowColor property, threeD property, topColor property, About colors and color references, Color Names Reference, Why don't buttons respect my color settings?, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies the color of a three-D object's lowered edge.

Syntax

set the bottomColor of object to {empty | colorName | RGBColor}

Example Code

```
set the bottomColor of scrollbar 1 to "purple"  
set the bottomColor of field 1 to the topColor of me
```

Comments

Objects whose threeD property is set to true appear to stick out of or recede into the screen. Use the bottomColor property to specify the color of the shadowed edge of the object.

Value:

The bottomColor of an object is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the bottomColor for all objects is empty.

Comments:

Setting the bottomColor of an object to empty allows the bottomColor of the object's owner to show through. Use the effective keyword to find out what color is used for the object, even if its own bottomColor is empty.

If the object's showBorder property is false, the bottomColor has no effect.

The setting of the bottomColor property has different effects, depending on the object type:

- ī The bottomColor of a stack determines the bottomColor of each object in the stack that does not have its own bottomColor.
- ī The bottomColor of a card or group determines the color of the border around the card or group, as well as determining the bottomColor of each object in the card or group that does not have its own bottomColor.
- ī The bottomColor of a button, player, EPS object, or graphic forms a border on the bottom and right edges of the object. If the object's threeD property is false, the bottomColor has no effect.
- ī The bottomColor of a field forms a border on the top and left edges of the field and (if the field is a scrolling field) the bottom and right edges of the arrow boxes at the ends of the scrollbar and the top and left edges of the scroll area. The field's bottomColor also determines the color of the bottom and right edges of any text in the field whose textStyle is set to "threeDBox". If the field's threeD property is false, the field border is not affected.
- ī The bottomColor of a scrollbar forms a border on the bottom and right edges of the arrow boxes at the ends of the scrollbar, and the top and left edges of the scroll area.
- ī The bottomColor of an audio clip or video clip has no effect.
- ī The bottomColor of an image is the sixth color in the image's color palette. (To set the color of the lowered edge of an image's border, set the bottomColor of the card, stack, or group that owns the image.)

If an object's bottomPattern is set, the pattern is shown instead of the color specified by the bottomColor.

bottomLeft

property

Synonyms

botLeft

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottom property, bottomRight property, height property, left property, location property, rectangle property, topLeft property, width property

Summary

Specifies the location of the specified object's lower left corner.

Syntax

set the bottomLeft of object to left,bottom

Example Code

```
set the bottomLeft of this stack to 400,250  
set the bottomLeft of button myButton to 100,the bottom of button 1
```

Comments

Use the bottomLeft property to change the placement of a control or window.

Value:

The bottomLeft of an object is any expression that evaluates to a pointótwo integers separated by a comma.

The first item of the bottomLeft is the distance in pixels from the left edge of the screen (for stacks) or card to the left edge of the object. The second item is the distance in pixels from the top edge of the screen (for stacks) or card to the bottom edge of the object.

For cards, the bottomLeft property is read-only and cannot be set.

Comments:

The bottomLeft of a stack is in absolute (screen) coordinates. The first item (the left) of a card's bottomLeft property is always zero; the second item (the bottom) is always the height of the stack window. The bottomLeft of a group or control is in relative (window) coordinates.

In window coordinates, the point 0,0 is at the top left of the stack window. In screen coordinates, the point 0,0 is at the top left of the screen.

Changing the `bottomLeft` of an object moves it to the new position without resizing it. To change an object's size, set its `height`, `width`, or `rectangle` properties.

Important! The order of the `bottom` and `left` parameters is reversed compared to the property name: `left` comes first, then `bottom`.

bottomMargin

property

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottom property, formattedHeight property, height property, leftMargin property, margins property, rightMargin property, topMargin property

Summary

Specifies how close text within an object can come to the object's lower edge, and how close objects in a group can come to the group's lower edge.

Syntax

set the bottomMargin of {button | field | group} to pixels

Example Code

```
set the bottomMargin of field "List" to 0
```

Comments

Use the bottomMargin property to change the amount of blank space between an object's bottom edge and its contents.

Value:

The bottomMargin is a non-negative integer.

By default, the bottomMargin of a newly created field is 8. If the field's wideMargins property is true, the field's bottomMargin is set to 14. The default bottomMargin setting for other controls is 4.

Comments:

The bottomMargin property of a field or button specifies how many blank pixels are left between the object's bottom edge and the lower edge of its text. The bottomMargin of a group specifies how far the group's bottom edge extends below its lowest object.

An object's bottomMargin property is equal to item 4 of its margins property.

bottomPattern

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

borderPattern property, bottomColor property, brushPattern property, effective keyword, focusPattern property, foregroundPattern property, hilitePattern property, patterns property, shadowPattern property, topPattern property

Summary

Specifies the pattern of a three-D object's lowered edge.

Syntax

set the bottomPattern of object to {empty | patternID | imageID}

Example Code

```
set the bottomPattern of scrollbar 2 to 4934
```

```
set the bottomPattern of button "OK" to savedPattern
```

Comments

Objects whose threeD property is set to true appear to stick out of or recede into the screen. Use the bottomPattern property to specify the color of the shadowed edge of the object.

Value:

The bottomPattern of an object is a pattern specifier.

A patternNumber is a built-in pattern number between 1 and 164. (These patterns correspond to Revolution's built-in patterns 136 to 300.)

An imageID is the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the bottomPattern for all objects is empty.

Comments:

Pattern images can be color or black-and-white.

Cross-platform note: To be used as a pattern on Mac OS systems, an image must be 128x128 pixels or less, and both its height and width must be a power of 2. To be used on Windows and Unix systems, height and width must be divisible by 8. To be used as a fully cross-platform pattern, both an image's dimensions should be one of 8, 16, 32, 64, or 128.

The `bottomPattern` of controls is drawn starting at the control's upper right corner: if the control is moved, the pattern does not shift.

Setting the `bottomPattern` of an object to empty allows the `bottomPattern` of the object's owner to show through. Use the `effective` keyword to find out what pattern is used for the object, even if its own `bottomPattern` is empty.

If the object's `showBorder` property is false, the `bottomPattern` has no effect.

The setting of the `bottomPattern` property has different effects, depending on the object type:

- ï The `bottomPattern` of a stack determines the `bottomPattern` of each object in the stack that does not have its own `bottomPattern`.
- ï The `bottomPattern` of a card or group determines the pattern of the border around the card or group, as well as determining the `bottomPattern` of each object in the card or group that does not have its own `bottomPattern`.
- ï The `bottomPattern` of a button forms a border on the bottom and right edges of the button. If the button's `threeD` property is false, the `bottomPattern` has no effect.
- ï The `bottomPattern` of a field forms a border on the top and left edges of the field and (if the field is a scrolling field) the bottom and right edges of the arrow boxes at the ends of the scrollbar and the top and left edges of the scroll area. The field's `bottomPattern` also determines the pattern of the bottom and right edges of any text in the field whose `textStyle` is set to "threeDBox". If the field's `threeD` property is false, the field border is not affected.
- ï The `bottomPattern` of a scrollbar forms a border on the bottom and right edges of the arrow boxes at the ends of the scrollbar, and the top and left edges of the scroll area.
- ï The `bottomPattern` of a graphic, audio clip, video clip, or image has no effect.
- ï The `bottomPattern` of a player or EPS object forms a border on the top and left edges of the object. If the object's `threeD` property is false, the `bottomPattern` has no effect.

If an object's `bottomPattern` is set, the pattern is shown instead of the color specified by the `bottomColor`.

bottomPixel

property

Synonyms

sixthPixel

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundPixel property, borderPixel property, bottomColor property, colorMap property, focusPixel property, foregroundPixel property, hilitePixel property, screenColors function, shadowPixel property, topPixel property, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies which entry in the color table is used for the color of a three-D object's lowered edge.

Syntax

set the bottomPixel of object to colorNumber

Example Code

```
set the bottomPixel of field "Status" to 14
```

Comments

Use the bottomPixel property to change the bottom color of an object when the bit depth of the screen is 8 bits (256 colors) or less.

Value:

The bottomPixel of an object is an integer between zero and (the screenColors - 1). It designates an entry in the current color table.

By default, the bottomPixel for all objects is empty.

Comments:

The bottomPixel property specifies which entry in the color table is used for an object's bottom color. It is similar to the bottomColor property, but is specified as an entry in the current color table, rather than as a color reference.

The color table can be set by changing the colorMap property.

bottomRight

property

Synonyms
botRight

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
bottom property, bottomLeft property, height property, location property, rectangle property, right property, topRight property, width property

Summary
Specifies the location of the specified object's lower right corner.

Syntax
set the bottomRight of object to right,bottom

Example Code
set the bottomRight of scrollbar "Master" to 400,200
set the bottomRight of field "Follower" to the mouseLoc

Comments
Use the bottomRight property to change the placement of a control or window.

Value:
The bottomRight of an object is any expression that evaluates to a point of two integers separated by a comma.

The first item of the bottomRight is the distance in pixels from the left edge of the screen (for stacks) or card to the right edge of the object. The second item is the distance in pixels from the top edge of the screen (for stacks) or card to the bottom edge of the object.

For cards, the bottomRight property is read-only and cannot be set.

Comments:
The bottomRight of a stack is in absolute (screen) coordinates. The first item (the right) of a card's bottomRight property is always the width of the stack window; the second item (the bottom) is always the height of the stack window. The bottomRight of a group or control is in relative (window) coordinates.

In window coordinates, the point 0,0 is at the top left of the stack window. In screen coordinates, the point 0,0 is at the top left of the screen.

Changing the bottomRight of an object moves it to the new position without resizing it. To change an object's size, set its height, width, or rectangle properties.

Important! The order of the bottom and right parameters is reversed compared to the property name: right comes first, then bottom.

boundingBox

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

postScript property, xExtent property, xOffset property, yExtent property, yOffset property

Summary

Specifies the value of the "%%BoundingBox" comment in an imported PostScript file.

Syntax

set the boundingBox of EPSObject to left,top,width,height

Example Code

```
set the boundingBox of EPS 1 to 100,100,500,600
```

Comments

Use the boundingBox property to control the appearance of an EPS object.

Value:

The boundingBox of an EPS object consists of four integers, separated by commas.

Comments:

The %%BoundingBox comment specifies the height, width, and placement on the page of a PostScript file. Use the boundingBox property to find out these properties of the PostScript code in an EPS object, or to set them if the PostScript file did not do so.

The four components of the boundingBox property are equal to the EPS object's xOffset, yOffset, xExtent, and yExtent properties respectively.

This property is supported only on Unix systems with Display PostScript installed.

boundingRect

property

Synonyms

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomMargin property, formattedHeight property, formattedWidth property, leftMargin property, rectangle property, rightMargin property, topMargin property, wideMargins property

Summary

Specifies whether a group automatically changes size to fit when its controls are moved or resized.

Syntax

set the boundingRect of group to {left,top,right,bottom | empty }

Example Code

```
set the boundingRect of group "Icons" to 100,100,500,500
set the boundingRect of the target to the rect of the target
```

Comments

Use the boundingRect property to control how a group responds when you move one of the group's controls to the edge of the group.

Value:

The boundingRect of a group consists of four integers separated by commas.

By default, the boundingRect property of a group is set to empty.

Comments:

If a group's boundingRect is empty and its lockLocation is false, when you drag an object toward the boundary of the group, the group automatically expands, resizing itself to fit. If the lockLocation is true, the object is clipped to the group's rectangle.

If a group's boundingRect is not empty and its lockLocation is false, when you drag an object toward the boundary of the group, the group does not automatically resize to fit its objects. Instead, the object is clipped at the boundingRect. (In group-editing mode, the entire control is shown, but when you exit group-editing mode, controls outside the boundingRect are clipped.)

If the group is a scrolling group, dragging an object in it automatically scrolls the group. When you drag beyond the scrollable area, the object is clipped.

box

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

borderColor property, plain keyword, textStyle property, threeDBox keyword, Text menu > Box

Summary

Used with the textStyle property to specify that a chunk of text in a field has a plain box drawn around it.

Syntax

Example Code

```
set the textStyle of the selectedChunk to box
```

Comments

Use the box keyword to emphasize text by drawing a box around it.

Comments:

The box is drawn in the object's borderColor and borderPattern.

If the text crosses more than one screen line, a box is drawn around each line of the text.

If the text has mixed styles, a separate box is drawn around each style run. For example, if three words are boxed and the last is also boldface, a box is drawn around the first two words, and another box is drawn around the boldface word.

An object or chunk of a field may have the box or threeDBox style, but not both at once.

break

control structure

Synonyms

exit switch

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

case keyword, end if keyword, exit control structure, switch control structure

Summary

Skips the rest of the current switch structure and goes to the statement following the end switch.

Syntax

break

Example Code

```
case thisVariable
switch true
  doSomething
  break
case false
  doSomethingElse
  break
end switch
```

Comments

Use the break control structure to end each case section in a switch structure.

Form:

The word break appears on a line by itself, within a case section of a switch control structure.

Comments:

A switch control structure consists of one or more conditions. For each condition, a set of statements is executed. The break control structure ends this set of statements and breaks out of the switch structure.

If no break appears, Revolution executes each set of statements in the following case sections, whether or not the condition in the case is true. This means that if you leave out the break control structure, more than one case in the switch statement may be executed. Occasionally, this is desirable, but usually you should include the break control structure at the end of each case, to ensure that only one set of statements is executed.

Note: The break control structure is implemented internally as a command and appears in the `commandNames`.

The break command should not be used outside a switch structure: such use is nonstandard, so it may not be apparent to whoever reads the code that the use is intentional. If it appears outside a switch structure, it has the same effect as the exit control structure.

break

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

break control structure, formfeed constant, print command

Summary

Used with the print command to skip to the next page.

Syntax

Example Code

```
print break
```

Comments

Use the break keyword to start a new page.

Comments:

The break keyword is most useful when printing a batch of cards with the open printing command, as it can be used to allocate cards across different pages as desired.

breakpoint

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

answer command, do command, Development menu > Script Debug Mode, Debug menu > Set Breakpoint (Script Editor)

Summary

Pauses the handler and displays the debugger window.

Syntax

breakpoint

Example Code

```
breakpoint  
if x > 0 then breakpoint
```

Comments

Use the breakpoint command to aid in debugging a handler. It is especially useful for checking to see whether values are staying in the correct range as the handler executes.

Comments:

The breakpoint command has no effect unless the stack is running in the Revolution development environment..

browse

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

choose command, pointer keyword, tool function, Tools menu > Browse Tool

Summary

Designates the Browse tool, which is used to interact with the objects of a stack.

Syntax

Example Code

```
choose browse tool
```

Comments

Use the browse keyword to enter normal (non-editing) mode.

Comments:

When using the Browse tool, the cursor is a pointing hand (or, over editable fields, an I-beam). This tool is used for user actions such as clicking buttons and typing in fields.

The Browse tool is always in effect for stacks whose `cantModify` property is true and for a stack being displayed as a palette, modeless dialog, or modal dialog, regardless of which tool is currently chosen.

brush

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

brush property, brushColor property, brushPattern property, choose command, tool function

Summary

Designates the paint tool used to draw freehand brushstrokes in an image.

Syntax

Example Code

```
choose brush tool
```

Comments

Use the brush keyword to paint brushstrokes with the brushColor.

Comments:

When using the Brush tool, the cursor is the shape specified by the brush property (over images) or an arrow (anywhere else). This tool is used to brush the brushColor (or brushPattern) onto an image.

If you try to paint with the brush on a card that has no images, an image the size of the card is created to hold the painting. If the current card already has one or more images, painting outside the image has no effect.

brush

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

brush keyword, brushColor property, brushPattern property, choose command, cursor property, eraser property, spray property, tool property

Summary

Specifies the shape used for painting with the Brush tool.

Syntax

set the brush to {brushID | imageID}

Example Code

```
set the brush to 13
```

```
set the brush to the short ID of image "My Custom Brush"
```

Comments

Use the brush property to specify which shape is painted by the Brush tool.

Value:

The brush is a brush specifier.

A brushID is a built-in brush number between 1 and 35. (These brushes correspond to Revolution's built-in patterns 101 to 135.)

An imageID is the ID of an image to use for painting with the brush. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the brush is set to 8 (a round brush).

Comments:

The entire area of the brush cursor is used as the brush shape. The shape painted by the brush is drawn in the brushColor, regardless of what colors might be in the image used for the brush shape.

When the Brush tool is in use, the cursor is the same as the brush shape. You can use any size image as a brush, but the cursor may appear distorted on some systems if the image is not 16x16 pixels.

Note: In order to use a brush, you must choose the Brush tool using either the Paint Tools palette or the choose command.

If you want to change the brush property in a standalone application—for example, to let the user paint in images—you must copy the stack "revCompatibilityBrushes1" into your application before you build the standalone. (This stack includes the cursors used for the brush tool.) To copy the stack, enter the following into the message box or a handler:

```
clone stack "revCompatibilityBrushes1"  
set the mainStack of this stack to "My Main Stack"  
-- (substitute your application's main stack name
```

A copy of "revCompatibilityBrushes1" is now a substack of your main stack, and will be saved in the stack file the next time you save the main stack.

Tip: If you want to create your own brush cursors, clone the "revCompatibilityBrushes1" stack and change the images in it to the desired shapes. Make sure to include the stack when you build the standalone.

brushColor

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundColor property, brush keyword, brush property, brushPattern property, bucket keyword, choose command, filled property, foregroundColor property, penColor property, spray can keyword, About colors and color references, Color Names Reference, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies the color used to paint and to fill shapes in an image.

Syntax

set the brushColor to {colorName | RGBColor}

Example Code

```
set the brushColor to "blue"  
set the brushColor to "#FFCC99"  
set the brushColor to 127,127,255
```

Comments

Use the brushColor property to change the color used with the bucket, spray can, and brush paint tools, and for the interior of filled shapes.

Value:

The brushColor is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the brushColor is black.

Comments:

If the `brushPattern` has been set since the last time the `brushColor` was set, the pattern is used instead of the color specified by `brushColor`. In other words, the last-set property takes priority.

brushPattern

property

Synonyms

pattern

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

brushColor property, choose command, filled property, penPattern property, tool property, How to add a pattern to the Image Chooser

Summary

Specifies the pattern used to paint with the Brush tool and to fill shapes in an image.

Syntax

set the brushPattern to {patternNumber | imageID}

Example Code

```
set the brushPattern to 204
```

Comments

Use the brushPattern property to change the pattern used with the bucket, spray can, and brush paint tools, and for the interior of filled shapes.

Value:

The brushPattern is a pattern specifier.

A patternNumber is a built-in pattern number between 1 and 164. (These patterns correspond to Revolution's built-in patterns 136 to 300.)

An imageID is the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the brushPattern is empty.

Comments:

Pattern images can be color or black-and-white.

Cross-platform note: To be used as a pattern on Mac OS systems, an image must be 128x128 pixels or less, and both its height and width must be a power of 2. To be used on Windows and Unix systems,

height and width must be divisible by 8. To be used as a fully cross-platform pattern, both an image's dimensions should be one of 8, 16, 32, 64, or 128.

If the `brushPattern` has been set since the last time the `brushColor` was set, the pattern is used instead of the color specified by `brushColor`. In other words, the last-set property takes priority.

bucket

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

brushColor property, brushPattern property, choose command, tool function

Summary

Designates the paint tool used to fill an area in an image.

Syntax

Example Code

```
choose bucket tool
```

Comments

Use the bucket keyword to fill a shape, or the entire image, with the brushColor.

Comments:

When using the Bucket tool, the cursor is a paintbucket shape (over images) or an arrow (anywhere else). This tool is used to fill shapes with a color. The clicked pixel, and all connected pixels of the same color, are changed to the brushColor (or brushPattern).

If you try to paint with the bucket on a card that has no images, an image the size of the card is created to hold the painting. If the current card already has one or more images, painting outside the image has no effect.

bufferHiddenImages

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

alwaysBuffer property, dontDither property, screenNoPmaps property

Summary

Specifies whether images that are not visible are decompressed into an offscreen buffer.

Syntax

set the bufferHiddenImages to {true | false}

Example Code

```
set the bufferHiddenImages to true
```

Comments

Use the bufferHiddenImages property to eliminate delays when hidden images are shown.

Value:

The bufferHiddenImages is true or false.

By default, the bufferHiddenImages is set to false.

Comments:

If the bufferHiddenImages property is set to true, hidden images are decompressed (along with visible images) when the user goes to a card. This means the image is shown immediately when its visible property is set to true, without waiting for decompression, but it uses more memory.

If the bufferHiddenImages is false, hidden images are not decompressed until they are made visible.

If the alwaysBuffer property of an image is true, the setting of the bufferHiddenImages property has no effect on that image, and it is always decompressed immediately.

bufferMode

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

colorWorld property, lockColorMap property, screenColors function, screenDepth function

Summary

Reports the number of colors the screen can display.

Syntax

get the bufferMode

Example Code

```
set the label of button "Color Mode" to the bufferMode
```

Comments

Use the bufferMode property to determine the color capacity of the screen. For example, you might display different images depending on the number of colors available.

Value:

The bufferMode property reports "millions" (for 32-bit or 24-bit color), "thousands" (for 16-bit color), or the number of colors.

This property is read-only and cannot be set.

Comments:

If the system has more than one monitor, the bufferMode property reports the number of colors displayed on the main screen.

The value reported by the bufferMode property is updated only when you start up the application. If you change the screen settings after starting up the application, you must quit and restart to update the bufferMode.

buildNumber

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

environment function, platform function, QTVersion function, systemVersion function, version function

Summary

Returns the build number of the Revolution engine.

Syntax

the buildNumber

buildNumber()

Example Code

```
the buildNumber
if the buildNumber < 3 then answer "This version is too early."
```

Comments

Use the buildNumber function to determine which engine version the application is using. This is useful to figure out, for example, whether a certain Transcript feature is available.

Value:

The buildNumber function returns an integer.

Comments:

The buildNumber of two copies of Revolution or a standalone application may be different, even if the version function returns the same number. This is because new builds of the same version may be released to provide fixes to specific problems.

When reporting a problem with Revolution, please include the buildNumber and version of the application.

busy
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, cursor property, six constant, watch constant

Summary

Equivalent to the number 6.

Syntax

Example Code

```
set the cursor to busy
```

Comments

Use the busy constant to set the cursor to a rotating beachball shape, suitable for showing the progress of lengthy operations.

Comments:

The busy cursor is a rotating beach ball. Each time you use the statement set the cursor to busy, the beach ball advances in its rotation. For example, the following statements cause the cursor to appear to spin as long as the repeat loop is running:

```
repeat until someCondition is true
  set the cursor to busy -- spins a bit further
  doSomething -- insert whatever you want the loop to do here
end repeat
```

The following two statements are equivalent:

```
set the cursor to busy
set the cursor to 6
```

However, the first is easier to read and understand in Transcript code.

Important! If you use the busy cursor or other standard cursors in your application, you must include it when you create your standalone. Make sure the "Cursors" option on the Inclusions tab in Step 3 of the Distribution Builder is checked.

button

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

choose command, create command, field keyword, graphic keyword, image keyword, player keyword, scrollbar keyword, style property, templateButton keyword, tool function

Summary

Designates the Button tool, which is used to create new buttons.

Syntax

Example Code

```
if the tool is not button then beep
```

Comments

Use the button keyword to create buttons.

Comments:

When using the Button tool, the cursor is a crosshairs. This tool is used for creating buttons by clicking at one corner of the button and dragging to the opposite corner.

You can set the style of the templateButton to the button type you want, then draw the button with the Button tool. The button icons on the Tools palette work this way: each one sets the style of the templateButton, then chooses the Button tool so you can create the button.

button

object

Synonyms

btn

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

templateButton keyword, About object types and object references, Object menu > New Control > Standard Button, Object menu > New Control > Rectangle Button, Object menu > New Control > Shadow Button, Object menu > New Control > Blank Button, Object menu > New Control > Check Box, Object menu > New Control > Radio Button, Object menu > New Control > Tabbed Button, Object menu > New Control > Pulldown Menu, Object menu > New Control > Popup Menu, Object menu > New Control > Option Menu, Object menu > New Control > Combo Box Menu

Summary

A control that is clickable.

Syntax

Example Code

```
set the hilite of button "Warning" to false
if the number of this card is 1 then hide button "Previous"
```

Comments

Use the button object type to create a clickable button, a tabbed window, or a menu.

Comments:

Button objects can be push buttons, checkboxes, radio buttons, or menus, depending on the setting of their style property.

A button whose style is set to "menu" can be a popup or contextual menu, pulldown menu, option menu, tabbed button, or combo box, depending on the setting of its menuMode property.

A button is contained in a card, group, or background. Buttons cannot contain other objects.

The button object has a number of properties and messages associated with it. To see a list of messages that can be sent to a button as a result of user actions or internal Revolution events, open the "Transcript Language Dictionary" page of the main Documentation window, and choose "Button Messages" from

the Show menu at the top. To see a list of all the properties a button can have, choose "Button Properties" from the Show menu.

by
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

combine command, divide command, each keyword, multiply command, rotate command, sort command, sort container command, split command

Summary

Used with the divide and multiply commands to specify a number; used with the rotate command to specify how far to rotate; used with the sort and sort container commands to specify the sort key; used with the combine and split commands to specify the delimiters for transforming a variable between an array and a string.

Syntax

Example Code

```
divide myVariable by 2  
rotate image "Calliope" by -45  
sort lines of field "Data" by item 2 of each
```

Comments

Use the by keyword to divide or multiply a container by a number or to specify how to sort.

Comments:

When used with the combine or split command, the by keyword is a synonym for using.

cachedURLs

function

Synonyms

cachedURL

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

load command, unload command, URL keyword, URLStatus function

Summary

Returns a list of the URLs that have been downloaded and copied to the cache using the load command.

Syntax

the cachedURLs

cachedURLs()

Example Code

```
the cachedURLs
```

```
if myURL is not among the lines of the cachedURLs then load myURL
```

Comments

Use the cachedURLs function to determine which files have been downloaded and are currently in the cache. A cached URL can be accessed more quickly than one that is not cached.

Value:

The cachedURLs function returns a list of currently loaded URLs, one per line.

Comments:

The cachedURLs function includes all URLs that you have successfully loaded with the load command. If an unsuccessful attempt has been made to load a URL, it does not appear in the cachedURLs. Only URLs whose URLStatus is "cached" appear in the cachedURLs.

Cached files consume memory. To release this memory after you are finished with a URL, use the unload command to remove it from the cache. You can use this handler to unload all URLs:

```
on unloadAll
  repeat for each line thisURL in the cachedURLs
    unload URL thisURL
  end repeat
```

end unloadAll

Important! The cachedURLs function is part of the Internet library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Changes to Transcript:

The URLStatus function became part of the Internet library in version 1.1. In previous versions, it was not a library function.

Starting with version 1.1.1, URLs that were not successfully downloaded do not appear in the cachedURLs. In previous versions, all URLs for which the load command was issued appeared in the cachedURLs, even if they were not successfully downloaded.

call

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

defaultStack property, do command, params function, pass control structure, send command, start using command, value function, About messages and the message path, How to call a custom function that's not in the message path, How to use a handler outside the message path

Summary

Executes the specified handler in any object's script.

Syntax

call handler [of object]

Example Code

```
call "mouseUp"  
call "doItAgain Sam,Ilsa" of card "Holding Pen"  
call myMessage of button "Help" of first card
```

Comments

Use the call command to use a handler that's not in the normal message path.

Parameters:

The handler is any handler in the script of the specified object, along with any parameters that handler requires, separated by commas. The entire handler including parameters must be enclosed in quotes.

The object is any object reference. If no object is specified, Revolution uses the current object.

Comments:

The call command sends a handler message to the object. If the script of the object doesn't trap the handler message, the message is passed to the next object in the object's message path.

Any object references in the handler you call are treated relative to the object whose script issued the call command, not relative to the object you specify. For example, button 3 refers to button 3 of the current card, not of the card where the object is located. In addition, if the handler contains a custom

command or custom function call, that command or function is sent to the original object, not to the object that contains the handler.

The difference between the call and send commands is that the send command changes the context so that object references are treated relative to the object you send the message to, while the call command does not change the context and continues to treat object references relative to the original object.

Tip: If a handler contains complex parameters, especially if some parameters contain double quotes, it may be simpler to put the message name and parameters into a variable, then use the name of the variable in the call statement.

callbacks

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentTime property, currentTimeChanged message, duration property, play command, playRate property, send command, timeScale property

Summary

Specifies messages to be sent during playback of a movie.

Syntax

set the callbacks of player to messageList

Example Code

```
set the callbacks of player "Mist" to myCallbacks  
set the callbacks of player "Do It!" to "2500,soundNotes"
```

Comments

Use the callbacks property to send callback messages to a player at certain points in the movie. This synchronizes the messages with the playback.

Value:

The callbacks of a player is a list of callbacks, one per line. Each callback consists of an interval number, a comma, and a message name.

By default, the callbacks property of newly created players is set to empty.

Comments:

When an interval number is reached during playback, Revolution sends the corresponding message to the player.

The number of intervals per second is specified by the player's timeScale property. The total number of intervals is given in the player's duration property.

cancel

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

pendingMessages function, send command, How to cancel a file transfer in progress, How to cancel a pending message, Recipe for 99 Bottles of Beer on the Wall (using send)

Summary

Removes a message that was queued with the send command and is waiting to be sent.

Syntax

cancel messageQueueID

Example Code

```
cancel 2298  
cancel item 1 of last line of the pendingMessages
```

Comments

Use the cancel command to get rid of messages that were set up but are no longer required.

Parameters:

The messageQueueID is the ID number of the message.

Comments:

The ID number of the message is returned by the send command that sent the message. This number is also the first item of the line corresponding to the message in the pendingMessages function.

All pending messages are automatically canceled when the application quits.

It is common to need to cancel a number of messages when leaving a card or stack, if the messages only pertain to that card or stack. For example, you might have queued a number of messages that create an animated display on the current card, and need to cancel them when the user goes to another card. The best solution in a case like this is to place each message ID number in a global variable or custom property at the time the message is sent. Then you can cancel all those messages in a repeat loop:

```
on closeCard
```

```
global myPendingMessages -- you've stored the message IDs here
repeat for each line thisMessageID in myPendingMessages
  cancel thisMessageID
end repeat
end closeCard
```

You can also cancel all pending messages with a similar handler, using the pendingMessages function:

```
on closeCard -- brute-force method
  repeat until the pendingMessages is empty
    cancel item 1 of line 1 of the pendingMessages
  end repeat
end closeCard
```

Caution! Revolution uses the send command to control Animation Builder animations and to control aspects of the development environment, so canceling Revolution's pending messages may cause unexpected behavior. All of Revolution's messages begin with "rev", so before canceling a pending message, if you're using the development environment, make sure the message name does not begin with "rev".

cantAbort

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

allowInterrupts property, cantModify property, How to prevent interrupting a handler, How to stop a running handler, Why can't I interrupt a handler?, Shortcut to stop a running handler

Summary

Specifies whether the user can halt a handler with a key combination.

Syntax

set the cantAbort of stack to {true | false}

Example Code

```
set the cantAbort of stack "Critical Functions" to true
```

Comments

Use the cantAbort property to prevent users from interrupting any of the handlers in a stack.

Value:

The cantAbort of a stack is true or false.

By default, the cantAbort property of newly created stacks is set to false.

Comments:

If a stack's cantAbort property is set to false, the user can halt a running handler in the stack by pressing Control-period or Control-break (on Windows or Unix) or Command-period (on Mac OS).

If the cantAbort is true, the user cannot interrupt a handler.

Caution! Before setting a stack's cantAbort property to true, make sure all handlers that may be affected have been thoroughly tested. If cantAbort is set to true, you cannot interrupt a runaway handler with the standard key combination.

cantDelete

property

Synonyms

Objects

group, card, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantModify property, delete command, delete stack command, deleteBackground message, deleteCard message, deleteGroup message, deleteStack message, destroyStack property

Summary

Specifies whether an object can be deleted.

Syntax

set the cantDelete of {card | group | stack} to {true | false}

Example Code

```
set the cantDelete of this card to true
```

Comments

Use the cantDelete property to protect a stack or part of a stack against accidental deletion.

Value:

The cantDelete of a card, background, or stack is true or false.

By default, the cantDelete property of newly created objects is set to false.

Comments:

If an object's cantDelete property is set to true, the object cannot be deleted either by user action or by a handler. If you want to delete the object, you must first set its cantDelete to false.

Deleting a card, background, or substack removes it permanently if the stack is saved after the deletion. Deleting a main stack removes it from memory, but does not remove its file from the user's system.

Setting a stack's cantDelete property to true does not prevent the user from deleting the file containing the stack by putting it in the Trash or Recycle Bin, or deleting it with a shell command or system script.

cantModify

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantDelete property, label property, lockText property, mode property, Shortcut to display a contextual menu

Summary

Specifies whether the user can make changes to a stack.

Syntax

set the cantModify of stack to {true | false}

Example Code

```
set the cantModify of stack "Help" to true
```

Comments

Use the cantModify property to protect the objects in a stack from changes.

Value:

The cantModify of a stack is true or false.

By default, the cantModify property of newly created stacks is set to false.

Comments:

If a stack's cantModify property is set to true, the user cannot choose any tool except the Browse tool. This prevents the user from moving, resizing, creating, or deleting objects.

If a stack's cantModify is false and its label property is empty, an asterisk (*) appears in the stack's title bar, indicating that the stack can be modified.

The cantModify property restricts user actions, but does not affect actions performed by a handler. To prevent either user action or a handler from deleting a card, group, or stack, use the cantDelete property.

Important! If a modifiable stack is open in an editable window, the modifiable stack takes precedence over any non-modifiable stacks, because its mode property is lower. This means that menu items (such

as Object menu Stack Properties) that act on the current stack may not be able to operate correctly with a stack whose `cantModify` is set to true as long as another, modifiable stack is open.

cantSelect

property

Synonyms

Objects

control

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

cantDelete property, disabled property, lockLocation property, select command, visible property

Summary

Specifies whether a control can be selected with the Pointer tool.

Syntax

set the cantSelect of object to {true | false}

Example Code

```
set the cantSelect of field "Background" to true
```

Comments

Use the cantSelect property to protect a control from being changed by the user.

Value:

The cantSelect of a control is true or false.

By default, the cantSelect of a newly created control is false.

Comments:

If the cantSelect of a control is set to true, the user cannot select it (and so cannot resize it or move it). When the user clicks the control with the Pointer tool, Revolution acts as though it had been clicked with the Browse tool. If the cantSelect is false, the user can click the control with the Pointer tool to select it.

A handler can still select the object, regardless of the setting of the cantSelect property.

capsLockKey

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

altKey function, commandKey function, controlKey function, keysDown function, metaKey function, optionKey function, shiftKey function

Summary

Returns the state of the Caps Lock key.

Syntax

the capsLockKey

capsLockKey()

Example Code

```
the capsLockKey
if the capsLockKey is down then exit repeat
```

Comments

Use the capsLockKey function to check whether the Caps Lock key is locked down.

Value:

The capsLockKey function returns down if the key is locked down and up if it's not.

card

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

card keyword, gray keyword, hide command, inverse keyword, show command, visual effect command, white keyword

Summary

Used with the visual effect command to show the destination card at the end of the visual effect. Also used to refer to cards.

Syntax

Example Code

```
visual effect push left to card  
go to card 1
```

Comments

Use the card keyword to improve the clarity of your code. For example, if a handler contains several visual effect commands, you can use the card keyword explicitly to point up the fact that different destination images are being used.

Comments:

Since by default, the image shown at the end of a visual effect is the destination card, you never actually need to use the card keyword; if you leave it out, the ending image is the card anyway.

card object

Synonyms
cd

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

templateCard keyword, About object types and object references, How to copy a card, How to find out which card is the current card, Object menu > New Card, Object menu > Delete Card

Summary

An object type that is a single page of a stack.

Syntax

Example Code

```
go to first card  
set the marked of this card to true
```

Comments

Use the card object type to display different sets of controls in the same stack window.

Comments:

A card corresponds to a single page of a stack: one card of each stack can be seen at a time. Each stack contains one or more cards.

Cards are contained in stacks, and may contain any kind of control.

The card object has a number of properties and messages associated with it. To see a list of messages that can be sent to a card as a result of user actions or internal Revolution events, open the "Transcript Language Dictionary" page of the main Documentation window, and choose "Card Messages" from the Show menu at the top. To see a list of all the properties a card can have, choose "Card Properties" from the Show menu.

cardIDs

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cardNames property, groupIDs property, ID property, How to list the cards in a stack

Summary

Reports the cards in a stack.

Syntax

get the cardIDs of stack

Example Code

```
get line 2 to 5 of the cardIDs of this stack  
repeat with y = 1 to the number of lines of the cardIDs of this stack
```

Comments

Use the cardIDs property to find out which cards are in the stack, or to find out which card IDs have already been used.

Value:

The cardIDs of a card or stack reports a list of short ID properties of cards, one per line.

This property is read-only and cannot be set.

cardNames

property

Synonyms

Objects

group, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cardIDs property, groupNames property, recentNames property, stackFiles property, substacks property,
How to list the cards in a stack

Summary

Lists the short name property of all the cards in a stack, or all the cards that contain a specified group.

Syntax

get the cardNames of {group | stack}

Example Code

```
put line 1 of the cardNames of this stack into firstCard  
go card (line x of the cardNames of group "Stats")
```

Comments

Use the cardNames property to list the cards in a stack.

Value:

The cardNames of a stack consists of a list of all the cards in the stack, one per line. The cardNames of a group consists of a list of all the cards that the group is placed on, one per line.

This property is read-only and cannot be set.

Comments:

Each line of the cardNames contains the short name of a card.

The cardNames of a group only reports card names for top-level groups (that is, groups that are not contained in another group). The cardNames property of a nested group always reports empty.

cascade

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

comboBox keyword, menuMode property, option keyword, popup keyword, pulldown keyword, style property, tabbed keyword, Object menu > New Control > Cascade Menu Item

Summary

Specifies one of the menu types that can be used with the menuMode property.

Syntax

Example Code

```
set the menuMode of button "Font" to cascade
```

Comments

Use the cascade keyword to create a hierarchical menu in a stack menu.

Comments:

A button whose style property is set to menu and whose menuMode property is set to cascade becomes a cascading menu. Each line of the button's contents is displayed as a menu item in the submenu.

case

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

break control structure, if control structure, switch control structure

Summary

Used within a switch control structure to designate one of the possibilities to be handled.

Syntax

case value

case condition

Example Code

```
case "A"  
case myChar = "A"
```

Comments

Use the case keyword to indicate the condition for which a set of statements should be executed.

Comments:

The case keyword can be used in two forms. The first form is used when the switch statement includes an expression: the case is executed if the value after the word case is equal to the expression in the switch control structure. For example, if the switch structure starts with this statement:

```
switch thisVariable
```

then a case statement inside the switch structure might look like this:

```
case 3
```

and the lines following the case keyword are executed if thisVariable = 3. Use this form if you want to test a single value, and do different things depending on what that value is equal to.

The second form is used when the switch statement does not include an expression: the case is executed if the condition after the word case is true. For example, if the switch structure starts with this statement:

switch

then a case statement inside the switch structure might look like this:

case the short name of this card contains "a"

and the statements following the case keyword are executed if the character "a" appears in the current card's name.

caseSensitive

property

Synonyms

Objects

local

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

= operator, find command, itemOffset function, lineOffset function

Summary

Specifies whether comparisons treat uppercase and lowercase letters as different.

Syntax

set the caseSensitive to {true | false}

Example Code

```
set the caseSensitive to true
```

Comments

Use the caseSensitive property to control the behavior of text comparisons.

Value:

The caseSensitive is true or false.

By default, the caseSensitive property is set to false.

Comments:

The caseSensitive property controls the behavior of the all string comparisons, including the comparison operators =, <>, <, <=, >, >=, is in, is among, is not among, and contains; the commands filter, find, and replace; and the functions offset, itemOffset, wordOffset, and lineOffset.

If the caseSensitive property is set to true, all the Transcript terms listed above, as well as all other string comparisons, treat uppercase and lowercase letters as different. For example, a search for "Apple" does not find the string "apple" or "APPLE", and the expression "APPLE" = "apple" evaluates to false.

If the caseSensitive is false, uppercase letters are treated as equal to their lowercase equivalents: a search for "Apple" finds strings such as "apple" or "APPLE" without paying attention to the case of each letter, and the expression "APPLE" = "apple" evaluates to true.

The `caseSensitive` also affects custom property names and array key names. If the `caseSensitive` is true, custom property names that differ only in the case of their letters are treated as different custom properties. If the `caseSensitive` is false, custom property names that differ only in letter case cannot be distinguished from each other. The same is true for the keys of array elements.

Since the `caseSensitive` is a local property, its value is reset to false when the current handler finishes executing. It retains its value only for the current handler, and setting it in one handler does not affect its value in other handlers it calls.

Important! Messages, object names, and Transcript terms are never treated as case-sensitive, even if the `caseSensitive` is set to true.

catch

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

errorDialog message, result function, throw control structure, try control structure

Summary

Used within a try control structure to handle errors.

Syntax

catch errorParameter

Example Code

```
catch myError
```

Comments

Use the catch keyword to handle errors returned by a handler with the throw control structure.

Comments:

When one of the statements in a try control structure causes an execution error, the errorParameter is set to the error string, and the statements after the catch keyword are executed. If Revolution generates the error (for example, an execution error from a built-in command), the errorParameter is a positive number. An error string can also be returned from a handler by the throw keyword.

The statements after the catch keyword are only executed if there is an error. These statements can refer to the value of the errorParameter. For example, the catch section might contain an if or switch control structure, which does different things depending on the value of the errorParameter.

centered

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

grid property, slices property, tool property

Summary

Specifies whether objects are drawn from the center to the edge, or from corner to corner.

Syntax

set the centered to {true | false}

Example Code

```
set the centered to true
```

Comments

Use the centered property to change the way objects are drawn when they are created with the object's tool, and to change how objects are resized.

Value:

The centered is true or false.

By default, the centered is set to false.

Comments:

If the centered property is set to true, when the user creates an object by clicking and dragging with the appropriate tool, the first point clicked is the center of the object, and the user drags to an edge to complete the object.

If the centered is false, the first point clicked is a corner, and the user drags to the opposite corner to complete the object.

The centered also affects how objects are resized with the Pointer tool. If the centered is true, dragging a handle at the corner or side of an object resizes the object symmetrically, so its center remains in the same place. If the centered is false, dragging a corner or side moves only that corner or side.

centuryCutoff

property

Synonyms

Objects

local

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

convert command, date function

Summary

Specifies which century two-digit years are assumed to be in.

Syntax

set the centuryCutoff to lastYearOfCentury

Example Code

```
set the centuryCutoff to 45
```

Comments

Use the centuryCutoff property to interpret dates close to the century mark.

Value:

The centuryCutoff is a two-digit number.

By default, the centuryCutoff is set to 35.

Comments:

The centuryCutoff specifies the two-digit year that ends the century. Two-digit years greater than the centuryCutoff belong to the next century; two-digit years less than or equal to the centuryCutoff belong to the previous century.

For example, if the centuryCutoff is set to 50, and the current year is 2000, then the date 4/12/51 is interpreted as being in the year 2051, while the date 4/12/50 is interpreted as being in 1950.

Since the centuryCutoff is a local property, its value is reset to 35 when the current handler finishes executing. It retains its value only for the current handler, and setting it in one handler does not affect its value in other handlers it calls.

character

keyword

Synonyms

char

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

item keyword, line keyword, token keyword, word keyword, About chunk expressions, Recipe for reversing a string

Summary

Designates a single character as part of a chunk expression.

Syntax

Example Code

```
get char 1 of field "Answers"  
put char 12 to 14 of it into oldAnswer
```

Comments

Use the character keyword to refer to a specific character or characters in a container.

Comments:

A character is a single letter, digit, punctuation mark, or control character. Spaces, tabs, and returns are all characters.

Important! Characters in chunk expressions are assumed to be single-byte characters. To successfully use chunk expressions with Unicode (double-byte) text, you must treat each double-byte character as a set of two single-byte characters. For example, to get the numeric value of the third Unicode character in a field, use statements like the following:

```
set the useUnicode to true -- makes charToNum assume Unicode  
get charToNum(char 5 to 6 of field "Chinese Text") -- gets bytes 5-6
```

characters

keyword

Synonyms

chars

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

character keyword, find command, is among operator, is not among operator, items keyword, lines keyword, number function, sort command, words keyword, About Chunk Expressions

Summary

Used with the sort command, number function, and is among and is not among operators to designate each character as a separate portion of a string. Also used with the find command to search for a string of characters.

Syntax

Example Code

```
get the number of chars of field "Entry"  
if "A" is among the characters of the bloodTypes then beep  
find characters "run" -- finds "run", "grunt", or "running"
```

Comments

Use the characters keyword to sort or select individual characters.

Comments:

A character is a single letter, digit, punctuation mark, or control character. Spaces, tabs, and returns are all characters.

When used with the find command, the characters keyword finds cards that contain the specified string of characters, wherever that string appears in a word.

charSet

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

charToNum function, ISOToMac function, macToISO function, numToChar function, platform function, textFont property

Summary

Indicates whether the Macintosh or ISO 8859 character set was used to enter a stack's text.

Syntax

get the charSet of stack

Example Code

```
if the charSet of this stack is "MacOS" then put "Mac" into thePlatform
```

Comments

Use the charSet property to determine which platform the stack was last saved on.

Value:

The charSet of a stack is "MacOS" or "ISO".

This property is read-only and cannot be set.

Comments:

If the charSet is "MacOS", the stack was last saved on a Mac OS system; if the charSet is "ISO", the stack was last saved on a Unix or Windows system.

When you open a stack on a Mac OS system that was last saved on a Unix or Windows system (or vice versa), the text in the stack is translated automatically to the appropriate character set. The process can take a perceptible amount of time, so it's a good idea to save a stack destined for a particular platform on that platform before delivering it to users.

The charSet property is changed for all stacks in the same stack file when the stack file is saved, so it is not possible for two stacks in the same file to have a different charSet.

charToNum

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

binaryDecode function, binaryEncode function, charSet property, ISOtoMac function, macToISO function, numToChar function, toLower function, toUpper function, uniEncode function, useUnicode property, Recipe for converting between characters and ASCII values

Summary

Returns the ASCII value of a character.

Syntax

the charToNum of character

charToNum(character)

Example Code

```
charToNum("A") -- returns 65
charToNum("ABC") -- also returns 65
if charToNum(nextChar) = 0 then next repeat
numToChar(charToNum("Z")) -- returns Z
```

Comments

Use the charToNum function to rank characters in their numerical order.

Parameters:

The character is any character or expression that evaluates to a character. If you specify a string containing more than one character, all characters but the first are ignored.

Value:

The charToNum function returns an integer between zero and 255. If the useUnicode property is set to true, the charToNum function returns an integer between zero and 65535.

Comments:

The charToNum function is the inverse of the numToChar function.

For special characters (those typed using the Option key or Alt key), the value returned by the charToNum function depends on the character set currently in use. On Mac OS systems this is normally

the Macintosh character set; on Unix systems, this is normally ISO 8859; on Windows systems, this is usually Code Page 1252, a variant of ISO 8859.

If the `useUnicode` property is set to `true`, you can specify a double-byte character. If the `useUnicode` is `false` and you specify a double-byte character, the `charToNum` function returns the numeric value of the character `div 256`.

Changes to Transcript:

The ability to handle Unicode characters was introduced in version 2.0. In previous versions, it was not possible to pass a Unicode character to the `charToNum` function.

checkmark

property

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

menuItem keyword, About menus and the menu bar, Recipe for checkmarking a menu item

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the checkmark of menuItem to {true | false}

Example Code

Comments

In HyperCard, the checkmark property determines whether a menu item has a checkmark next to it.

A handler can set the checkmark to any value without causing a script error, but the actual checkmark is not changed.

choose

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

browse keyword, brush keyword, bucket keyword, button keyword, click command, curve keyword, drag command, dropper keyword, eraser keyword, field keyword, graphic keyword, image keyword, line keyword, newTool message, oval keyword, pencil keyword, player keyword, pointer keyword, polygon keyword, rectangle keyword, regular keyword, roundRect keyword, scrollbar keyword, select keyword, spray can keyword, tool function, tool property, topLevel command, Tools menu > Browse Tool, Tools menu > Pointer Tool, Tools menu > Tools Palette, Shortcut to switch between browse and pointer tool

Summary

Selects a tool or paint tool.

Syntax

choose toolName tool

Example Code

```
choose browse tool  
choose savedTool tool
```

Comments

Use the choose command to select objects, create objects, or paint under scripted control.

Parameters:

The toolName is one of the basic tools from the Tools palette, or one of the paint tools from the Paint palette. The basic tools are:

- browse: perform actions such as clicking buttons and typing in fields
- pointer: select, move, or resize objects
- button: create all styles of button objects
- field: create all styles of field objects
- scrollbar: create all styles of scrollbar objects
- graphic: create all styles of graphic objects
- image: create paint images
- player: create sound and video player objects

The paint tools are:

- select: select a rectangle in an image
- pencil: draw freehand in an image
- bucket: fill a shape in an image
- brush: draw brushstrokes in an image
- eraser: erase an area in an image
- spray can: draw airbrush strokes in an image
- rectangle: draw a rectangle or square shape
- line: draw a straight line in an image
- round rect: draw a rectangle with rounded corners in an image
- oval: draw an oval or circle shape in an image
- polygon: draw a polygon in an image
- curve: draw a curved line in an image
- regular polygon: draw a regular polygon shape in an image
- dropper: pick up a color from an image

Comments:

The tool you choose is effective for all editable windows, until you choose another tool. (The Browse tool is always in effect for stacks whose `canModify` property is true, and for stacks being displayed as a palette, modeless dialog, or modal dialog.)

All types of graphics are drawn with the single Graphic tool. To draw a specific shape, first draw with the Graphic tool, then set the style property of the new graphic to the shape you want. (You can also set the style of the `templateGraphic` to the shape you want, then draw the graphic with the Graphic tool. The graphic icons on the Tools palette work this way: each one sets the style of the `templateGraphic`, then chooses the Graphic tool.)

Similarly, all types of buttons, fields, and scrollbars can be drawn with the single tool for that object type. You specify a particular kind of button, field, or scrollbar by setting its style property.

Note: You must choose tools by name. The HyperCard method of choosing tools by number is not supported in Revolution.

clear

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of graphic "Overlay" to clear
```

Comments

Use the clear keyword to turn an object black.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

When the clear mode is used, each component of the color underneath the object (red, green, and blue) is lowered to be equal to black.

The clear mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear as though their ink were set to reverse.

click

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

clickLoc function, drag command, dragStart message, mouseDown message, mouseLoc function, mouseUp message, send command, type command, How to find out what text the user clicked, How to find out where the user clicked, How to simulate a mouse click

Summary

Simulates a mouse click.

Syntax

click [button mouseButtonNumber] at point [with key [,key [,key]]]

Example Code

```
click at "100,200"  
click button 2 at the loc of card button "Start"  
click at the clickLoc
```

Comments

Use the click command to simulate the action of a click, instead of sending a mouseDown or mouseUp message—for example, to create a graphic or image with a handler.

Parameters:

The mouseButtonNumber is the number of a mouse button:

- 1 is the mouse button on Mac OS systems and the left button on Windows and Unix systems.
- 2 is the middle button on Unix systems.
- 3 is the right button on Windows and Unix systems, or Control-click on Mac OS systems.

If you don't specify a mouseButtonNumber, button 1 is used.

The point is any expression that evaluates to a point—a vertical and horizontal distance from the top left of the current stack, separated by a comma.

The key is one of `commandKey`, `controlKey`, `optionKey`, or `shiftKey`. You can specify up to three keys, separated by commas. (On Windows and Unix, `commandKey` indicates the Control key.)

Comments:

The click command sends a `mouseDown` and `mouseUp` message to the object at the clicked location. If two objects both occupy the clicked location—one overlapping the other—the messages are sent to the object on top.

If the object under the point is a button whose `autoHilite` is true, the click command causes the button to highlight and unhighlight, just as though the user had clicked.

If the point is not within one of the application's windows, the click command has no effect. This means that you cannot use the click command to switch to another application.

clickChar

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

charToNum function, clickCharChunk function, clickField function, clickLoc function, clickText function, lockText property

Summary

Returns the character the user last clicked in a field.

Syntax

the clickChar
clickChar()

Example Code

```
the clickChar  
if the clickChar is not "*" then beep 2
```

Comments

Use the clickChar function within a mouseDown, mouseUp, or selectionChanged handler to determine which character the user clicked, in order to provide hypertext (clickable text) or take some action based on the clicked character.

Value:

The clickChar function returns a character.

Comments:

The clickChar function is cleared at the next mouse click, as well as after some editing actions such as deleting text. If the last click was not in a field, the clickChar function returns empty.

If the field is locked, the clickChar function is most useful within a handler (such as mouseDown or mouseUp) that is triggered by the mouse click.

If the field is unlocked, mouseDown and mouseUp messages are not sent when the user clicks in the field (unless the user right-clicks or holds down the Control key while clicking). Use the clickChar

function within a `selectionChanged` handler to determine what characters the user is editing in an unlocked field.

To find the position of the `clickChar`, use the `clickChunk` function.

clickCharChunk

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

charToNum function, clickChar function, clickField function, clickLoc function, clickText function, foundChunk function, mouseCharChunk function, selectedChunk function

Summary

Returns the position of the character the user last clicked in a field.

Syntax

the clickCharChunk

clickCharChunk()

Example Code

```
the clickCharChunk
```

```
if word 2 of the clickCharChunk < 100 then put the clickChar into me
```

Comments

Use the clickCharChunk function within a mouseDown, mouseUp, or selectionChanged handler to determine where in the text the user clicked, in order to provide hypertext (clickable text) or take some action based on the click.

Value:

The clickCharChunk function returns a chunk expression of the form char charNumber to charNumber of field fieldNumber.

Comments:

The clickCharChunk function is cleared at the next mouse click, as as after some editing actions such as deleting text. If the last click was not in a field, the clickCharChunk function returns empty.

The charNum is the character the mouse pointer was over when the mouse was clicked. Moving the mouse before the mouse button is released does not affect the value returned by the clickCharChunk.

The first and second character numbers in the return value are always identical, unless the click was on a field but there was no text under it. In this case, the `clickCharChunk` returns a chunk expression of the form

`char charNumber to charNumber - 1 of field fieldNumber`
indicating the start of the `clickLine`. For example, if the mouse is over an empty field, the `clickCharChunk` returns `char 1 to 0 of field fieldNumber`.

If the field is locked, the `clickCharChunk` function is most useful within a handler (such as `mouseDown` or `mouseUp`) that is triggered by the mouse click.

If the field is unlocked, `mouseDown` and `mouseUp` messages are not sent when the user clicks in the field (unless the user right-clicks or holds down the Control key while clicking). Use the `clickCharChunk` function within a `selectionChanged` handler to determine what characters the user is editing in an unlocked field.

To get the actual character clicked, use the `clickChar` function.

clickChunk

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickChar function, clickCharChunk function, clickField function, clickLine function, clickLoc function, clickText function, foundChunk function, mouseChunk function, selectedChunk function

Summary

Returns the position of the word or grouped text that the user last clicked.

Syntax

the clickChunk

clickChunk()

Example Code

```
the clickChunk  
select the clickChunk
```

Comments

Use the clickChunk function within a mouseDown, mouseUp, or selectionChanged handler to determine which word or text group the user clicked, in order to provide hypertext (clickable text) or take some action based on the click.

Value:

The clickChunk function returns a chunk expression of the form char startChar to endChar of field fieldNumber.

Comments:

The clickChunk function is cleared at the next mouse click, as well as after some editing actions such as deleting text. If the last click was not in a field, the clickChunk function returns empty.

The returned value reports the word the user clicked: the startChar is the first character of the word, and the endChar is the last character. If the textStyle of the clicked text is "link", the returned value specifies the entire text group.

Important! Words are defined a little differently by the `clickChunk` function than the way they are used in chunk expressions. A word, for purposes of the `clickChunk`, is any text delimited by spaces, tabs, returns, or punctuation. If you click a punctuation character, only that character is returned. This means that, for example, clicking a hyphenated word only returns the part of the word that was clicked.

The returned value indicates the text that the mouse pointer was over when the mouse was clicked. Moving the mouse before the mouse button is released does not affect the value returned by the `clickChunk`.

If the field is locked, the `clickChunk` function is most useful within a handler (such as `mouseDown` or `mouseUp`) that is triggered by the mouse click.

If the field is unlocked, `mouseDown` and `mouseUp` messages are not sent when the user clicks in the field (unless the user right-clicks or holds down the Control key while clicking). Use the `clickChunk` function within a `selectionChanged` handler to determine what characters the user is editing in an unlocked field.

To get the text of the word or text group clicked, use the `clickText` function.

clickField

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickChunk function, clickText function, foundField function, mouseControl function, number property, selectedField function

Summary

Returns the number of the last field the user clicked.

Syntax

the clickField

clickField()

Example Code

```
the clickField
if the clickField is field "New Items" then exit mouseUp
```

Comments

Use the clickField function within a mouseDown, mouseUp, or selectionChanged handler to determine which field the user clicked, in order to provide hypertext (clickable text) or take some action based on the click.

Value:

The clickField function returns the number of the clicked field.

Comments:

The clickField function is cleared at the next mouse click, as well as after some editing actions such as deleting text. If the last click was not in a field, the clickField function returns empty.

The returned value indicates the field that the mouse pointer was over when the mouse was clicked. Moving the mouse before the mouse button is released does not affect the value returned by the clickField.

If the field is locked, the clickField function is most useful within a handler (such as mouseDown or mouseUp) that is triggered by the mouse click.

If the field is unlocked, `mouseDown` and `mouseUp` messages are not sent when the user clicks in the field (unless the user right-clicks or holds down the Control key while clicking). Use the `clickField` function within a `selectionChanged` handler to determine what field the user is editing.

clickH

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clickLoc function, clickV function, mouseH function

Summary

Returns the horizontal position of the most recent mouse click.

Syntax

the clickH
clickH()

Example Code

```
the clickH  
put the clickH div 2 into whichHalf
```

Comments

Use the clickH function to find out where the user last clicked.

Value:

The clickH function returns a non-negative integer.

Comments:

The returned value is the horizontal distance in pixels from the left edge of the clicked stack to the location of the click. (Use the clickStack function to identify which stack was clicked.)

The click location is the position of the mouse pointer when the user released the mouse button, not when the user pressed the button. If the user clicks and drags, then releases the mouse, the clickH reflects the mouse's position after the drag.

This function is equal to item 1 of the clickLoc function.

clickLine

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickChunk function, clickField function, clickStack function, clickText function, foundLine function, number property, selectedLine function

Summary

Returns the number of the line that the user last clicked in a field.

Syntax

the clickLine

clickLine()

Example Code

```
select the clickLine  
put the value of the clickLine into textOfClickedLine
```

Comments

Use the clickLine function within a mouseDown, mouseUp, or selectionChanged handler to determine which line the user clicked, in order to provide hypertext (clickable text) or take some action based on the click.

Value:

The clickLine function returns a chunk expression of the form line lineNumber of field fieldNumber.

Comments:

The clickLine function is cleared at the next mouse click, as well as after some editing actions such as deleting text. If the last click was not in a field, the clickLine function returns empty.

The returned value indicates the line that the mouse pointer was over when the mouse was clicked. Moving the mouse before the mouse button is released does not affect the value returned by the clickLine.

If the field is locked, the clickLine function is most useful within a handler (such as mouseDown or mouseUp) that is triggered by the mouse click.

If the field is unlocked, `mouseDown` and `mouseUp` messages are not sent when the user clicks in the field (unless the user right-clicks or holds down the Control key while clicking). Use the `clickLine` function within a `selectionChanged` handler to determine what line the user is editing in an unlocked field.

To get a chunk expression describing the word or text group that was clicked, use the `clickChunk` function.

clickLoc

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clickH function, clickV function, mouseLoc function, How to find out where the user clicked

Summary

Returns the position of the most recent mouse click.

Syntax

the clickLoc
clickLoc()

Example Code

```
the clickLoc  
if the clickLoc is the mouseLoc then send mouseUp to me
```

Comments

Use the clickLoc function to determine where a user clicked, or to determine whether the mouse pointer has moved since the last click.

Value:

The clickLoc function returns two integers separated by a comma.

Comments:

The first item of the returned value is the horizontal distance in pixels from the left edge of the clicked stack to the location of the click. (Use the clickStack function to identify which stack was clicked.) The second item of the returned value is the vertical distance from the top edge of the clicked stack to the location of the click.

The click location is the position of the mouse pointer when the user released the mouse button, not when the user pressed the button. If the user clicks and drags, then releases the mouse, the clickLoc reflects the mouse's position after the drag.

The first item of the clickLoc is equal to the clickH. The second item is equal to the clickV.

clickStack

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backdrop property, clickCharChunk function, clickChunk function, clickField function, clickLine function, clickLoc function, defaultStack property, name property

Summary

Returns the name of the stack in which the user most recently clicked the mouse.

Syntax

the clickStack

clickStack()

Example Code

```
the clickStack
repeat until (the short name of the clickStack is "Icon Palette")
if the name of the clickStack is not the defaultStack then beep
```

Comments

Use the clickStack function to determine which window the user clicked, or, in conjunction with the clickField, clickLine, or clickChunk, to determine which field was clicked.

Value:

The clickStack function returns the long name property of the stack that was last clicked.

Comments:

If the user clicks a stack's title bar or border to bring it to the front, the value returned by the clickStack does not change. Only clicking in the stack window's content area changes the clickStack value.

If the user clicks the backdrop, the clickStack returns empty.

clickText

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clickChar function, clickChunk function, clickStack function, foundText function, mouseText function, selectedText function

Summary

Returns the word or text group the user last clicked.

Syntax

the clickText
clickText()

Example Code

```
the clickText  
if the clickText contains "*" then goToFootnote (the clickText)
```

Comments

Use the clickText function within a mouseDown, mouseUp, or selectionChanged handler to determine which word or text group the user clicked, in order to provide hypertext (clickable text) or take some action based on the click.

Value:

The clickText function returns the text that the user clicked.

Comments:

The clickText function is cleared at the next mouse click, as well as after some editing actions such as deleting text. If the user has clicked anywhere since clicking the field, the clickText function returns empty.

The returned value contains the word the user clicked. If the textStyle of the clicked text is "link", the returned value contains the entire text group.

Important! Words are defined a little differently by the clickText function than the way they are used in chunk expressions. A word, for purposes of the clickText, is any text delimited by spaces, tabs,

returns, or punctuation. If you click a punctuation character, only that character is returned. This means that, for example, clicking a hyphenated word only returns the part of the word that was clicked.

The returned value indicates the text that the mouse pointer was over when the mouse was clicked. Moving the mouse before the mouse button is released does not affect the value returned by the `clickText`.

If the field is locked, the `clickText` function is most useful within a handler (such as `mouseDown` or `mouseUp`) that is triggered by the mouse click.

If the field is unlocked, `mouseDown` and `mouseUp` messages are not sent when the user clicks in the field (unless the user right-clicks or holds down the Control key while clicking). Use the `clickText` function within a `selectionChanged` handler to determine what text the user is editing in an unlocked field.

To get the location of the word or text group clicked, use the `clickChunk` function.

clickV

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clickH function, clickLoc function, mouseV function

Summary

Returns the vertical position of the most recent mouse click.

Syntax

the clickV
clickV()

Example Code

```
the clickV  
if the clickV > the bottom of stack window then beep -- below window
```

Comments

Use the clickV function to find out where the user last clicked.

Value:

The clickV function returns a positive integer.

Comments:

The returned value is the vertical distance in pixels from the top edge of the clicked stack to the location of the click. (Use the clickStack function to identify which stack was clicked.)

The click location is the position of the mouse pointer when the user released the mouse button, not when the user pressed the button. If the user clicks and drags, then releases the mouse, the clickV reflects the mouse's position after the drag.

This function is equal to item 2 of the clickLoc function.

clipboard

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clipboardData property, copy command, cut command, paste command

Summary

Returns the type of information on the clipboard.

Syntax

the clipboard
clipboard()

Example Code

```
if the clipboard is "text" then paste  
if the clipboard is empty then answer "Nothing to paste!"
```

Comments

Check the clipboard function before using the paste command, to ensure that you're pasting the right kind of data.

Value:

The clipboard function returns text, image, objects, or empty.

Comments:

The value of the clipboard function is a string describing what kind of data is on the clipboard. Revolution can paste text that has been cut or copied, or picture content from an image, one or more objects that have been cut or copied.

If the clipboard has nothing in it, or the contents of the clipboard is a type of data that is not text, image data, or objects, the clipboard function returns empty.

Tip: To get or change the data on the clipboard, use the clipboardData property.

Changes to Transcript:

The ability to sense image data on the clipboard was added in version 2.0.

clipboardData

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clipboard function, copy command, cut command, paste command, How to copy an image to the clipboard, Recipe for collecting text selections on the clipboard

Summary

Specifies what data and of what type is on the clipboard.

Syntax

set the clipboardData to clipboardArray

Example Code

```
set the clipboardData["text"] to "Meep!"  
set the clipboardData["RTF"] to the RTFText of field "Info"  
set the clipboardData["image"] to image 1
```

Comments

Use the clipboardData property to put data in the clipboard in a specified format, or to get the contents of the clipboard, without copying or pasting.

Value:

The clipboardData is an array with one or more of the following elements:

text	The text on the clipboard
HTML	The styled text on the clipboard, in the same format as the htmlText
RTF	The styled text on the clipboard, in the same format as the RTFText
Unicode	The text on the clipboard, in the same format as the unicodeText
image	The data of an image (in PNG format)
files	The name and location of the file or files on the clipboard, one per line

If the clipboard does not contain data of the specified type, the array element for that type is empty. (For example, if the clipboard contains text, clipboardData["image"] is empty.)

Comments:

The clipboardData property is populated automatically when the clipboard content is changed, either by using the cut or copy command, or by cutting or copying in another application.

You can use the clipboardData property to get the data on the clipboard. For example, if the clipboard contains styled text, you can put that text (in htmlText format) in a variable with the following statement:

put the clipboardData["htmlText"] into myVariable

To change the contents of the clipboard, you can set the clipboardData property directly. For example, the following statement places the text "Hello World" on the clipboard:

set the clipboardData["text"] to "Hello World"

The above statement is equivalent to selecting the text "Hello World" in a field and choosing Edit menu Copy, or to using the copy command. The data you place on the clipboard is accessible to your application using the paste command, and to other applications (that support pasting text) using the application's Paste menu item.

Tip: To quickly find out what kind of data is on the clipboard, use the clipboard function.

clone

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

copy command, create command, save command, paste command, How to duplicate an object, Why is a control the wrong size when created by a handler?, Edit menu > Duplicate, Edit menu > Replicate..., Shortcut to duplicate a control

Summary

Duplicates an object.

Syntax

clone object

Example Code

```
clone the selectedObject
clone player "Hard Day's Night"
clone this card
```

Comments

Use the clone command to create a copy of an existing object.

Parameters:

The object is any object reference.

Comments:

If the object is a control, its copy is placed on the current card, 32 pixels below and to the right of the original object. The copy's name and other properties are the same as those of the original. If the object is a grouped control, the clone is also owned by the group.

When the new control is created, the Pointer tool is automatically chosen. If you use the clone command in a handler, you can use the following statement after the clone command to resume using the Browse tool:

```
send "choose browse tool" to me in 1 tick
```

If the object is a card, the copy becomes the current card.

If the object is a stack, the newly created stack is opened. It is named "Copy of" and the stack's name.

The clone command places the name property of the newly created object in the it variable.

The clone command does not affect the contents of the clipboard.

close

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close file command, close printing command, close process command, close socket command, destroyStack property, go command, revCloseVideoGrabber command, How to open and close a drawer, How to respond to closing a window, File menu > Close

Summary

Closes a stack window.

Syntax

close stack

Example Code

```
close stack "Fearsome Stack"  
close the defaultStack
```

Comments

Use the close command to close a stack without user intervention.

Parameters:

The stack is any open stack.

Comments:

The close command closes the stack window immediately, but it does not necessarily remove the stack from memory. A stack that is closed but in memory still takes up space in RAM and its objects are still accessible to scripts.

If the stack's destroyStack property is set to false, or there are other stacks in the same file that are still open, closing the stack does not purge the stack from memory.

If the handler that closes the stack is in the script of the stack (or in the script of an object in the stack) and the stack's destroyStack property is true, the stack window is closed immediately, but the stack is not removed from memory until after the handler completes.

close driver

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close file command, driverNames function, open driver command, read from driver command, write to driver command

Summary

Closes a device driver that was opened with the open driver command.

Syntax

close driver deviceName

Example Code

```
close driver "USBCam1"  
close driver (line 2 of it)
```

Comments

Use the close driver command after you're finished communicating with a peripheral device.

Parameters:

The deviceName is the name of a device driver that's installed on the system and that you have previously opened with the open driver command.

Comments:

Any device drivers you have opened are closed automatically when you quit the application.

If you try to close a driver that is not already open, the result function is set to File is not open.

Changes to Transcript:

Support for using serial drivers with OS X systems was added in version 2.0.

close file

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

open file command, openFiles function, read from file command, sysError function, write to file command, About filename specifications and file paths

Summary

Closes a file that was opened with the open file command.

Syntax

close file filePath

Example Code

```
close file "/Desktop/Temp"  
close file it
```

Comments

Use the close file command to release a file after reading from it or writing to it. (A file that is open for writing can't be used by any other application until you've closed the file.)

Parameters:

The filePath is the name and location of the file you want to close. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

Comments:

Any files you have opened are closed automatically when you quit the application.

If you try to close a file that is not already open, the result function is set to File is not open.

close printing command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

open printing command, print command

Summary

Sends the current print job to the printer.

Syntax

close printing

Example Code

```
close printing  
if the number of this card > 10 then close printing
```

Comments

Use the close printing command to print cards that you've selected with the print card command.

Comments:

If you use print after the open printing command, the cards you print are saved up to be printed as a single batch. The close printing command prints out this batch of cards, enabling you to print multiple cards on a page.

If no print card commands have been executed since the open printing command, the close printing command prints a blank page. If you used the open printing with dialog form of the open printing command, and the user canceled the dialog, you don't need to issue a close printing command.

close process

command

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

kill command, open process command, openProcesses function, read from process command, write to process command

Summary

Closes a process that was started with the open process command.

Syntax

close process processName

Example Code

```
close process "/bin/sh"  
close process myOpenProgram
```

Comments

Use the close process command to tell a process to exit after you've finished using it.

Parameters:

The processName is the file path to the process you opened with the open process command.

Comments:

On Unix and Windows systems, close process closes the process's standard input. The process then finishes processing whatever data remains, and exits when done. On Mac OS systems, the close process command has no effect; you can use the kill command instead to quit an application that Revolution launched.

The process takes a short time to exit after you issue the close process command. Since two processes cannot have the same name, you need to wait for a process to exit before opening a new process with the same name. To reopen the same process after closing it, use the wait command to delay until the process has finished exiting:

```
close process myProcess  
wait until myProcess is not among the lines of the processNames
```


`open process myProcess`

If a process was opened with the access mode `neither`, it exits automatically when it finishes running, and does not need to be closed.

Changes to Transcript:

Support for using the `close process` command on OS X systems was added in version 2.0.

close socket command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

hostNameToAddress function, open socket command, openSockets function, read from socket command, resetAll command, write to socket command, How to unwedge upload and download operations

Summary

Shuts down a connection that was opened with the open socket or accept command.

Syntax

close socket socketID

Example Code

```
close socket "127.0.0.0"  
close socket "www.example.net:8080|newConnection"  
close socket (line 1 of the openSockets)
```

Comments

Use the close socket command to release the connection when you're finished getting and sending data over it.

Parameters:

The socketID is the identifier (set when you opened the socket) of the socket you want to close.

The socket identifier starts with the IP address of the host the socket is connected to, and may optionally include a port number (separated from the IP address by a colon). If there is more than one socket connected to that host and port, you can specify which socket by appending the connection name or number that was assigned when the socket was opened, separated from the port number by a vertical bar (|).

closeBackground

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close command, closeCard message, closeStack message, go command, openBackground message

Summary

Sent to the current card when the user is leaving a card that has a group to go to one that doesn't have the group.

Syntax

closeBackground backgroundID

Example Code

```
on closeBackground theClosedGroupID
  -- save when exiting a particular background:
  if theClosedGroupID is 234 then saveNotesToFile
  pass closeBackground
end closeBackground
```

Comments

Handle the closeBackground message if you want to perform cleanup or do other tasks when the user leaves a background.

Parameters:

The backgroundID is the ID number of the background being exited.

Comments:

A background is closed when the user either goes to a card that doesn't have the group on it, or closes the stack.

The closeBackground message is sent only if the group's backgroundBehavior property is set to true. If the group's backgroundBehavior is false, no closeBackground message is sent, even if the group is placed on multiple cards.

A separate closeBackground message is sent for each background that is being closed.

closeBox

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeStack message, closeStackRequest message, collapseBox property, decorations property, zoomBox property

Summary

Shows a window's close box.

Syntax

set the closeBox of stack to {true | false}

Example Code

```
set the closeBox of this stack to false
```

Comments

Use the closeBox property to display the close box in a window's title bar.

Value:

The closeBox property of a stack is true or false. By default, the closeBox of a newly created stack is set to true.

Comments:

The setting of this property affects the decorations property, and vice versa. Setting a stack's closeBox property determines whether its decorations property includes "close" (or is "default", for window styles that normally include a close box). Conversely, setting a stack's decorations property sets its closeBox to true or false depending on whether the decorations includes "close" (or is "default").

Note: On OS X systems, if the closeBox property is false, the close box is disabled rather than hidden.

Setting the closeBox property causes the stack window to flash briefly.

Changes to Transcript:

The closeBox property was fully implemented in version 2.1. In previous versions, it was included in Transcript for compatibility with imported SuperCard projects, but setting it had no effect.

closeCard

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close command, closeBackground message, closeStack message, go command, openCard message, preOpenCard message

Summary

Sent to the current card when the user goes to another card.

Syntax

closeCard

Example Code

```
on closeCard -- record date and time the card was last viewed
  -- in a custom property of the card
  set the lastAccessDate of the target to the seconds
  pass closeCard
end closeCard
```

Comments

Handle the closeCard message if you want to perform cleanup or do other tasks when the user leaves a card.

Comments:

A card is closed when the user either goes to another card in the same stack, or closes the stack.

The actual navigation is not triggered by the closeCard message, so trapping the message and not allowing it to pass does not prevent the card from closing.

closeField

message

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

exitField message, openField message, select command, How to respond to a change in field contents, Recipe for storing a modification timestamp

Summary

Sent to a field when the focus is being removed from that field and the field's content has changed.

Syntax

closeField

Example Code

```
on closeField -- make sure the user has entered a valid date
  if the text of me is not a date then
    beep
    answer "Please enter a date."
    select text of me
  end if
end closeField
```

Comments

Handle the closeField message if you want to make sure a field's content is correct after it has been changed.

Comments:

The selection is removed from a field (and the field loses focus) when another window is brought to the front, when the user clicks in another field, or when the user tabs out of the field. The field also loses focus when the select command is used to select text in another field. However, the closeField message is not sent when the user clicks in the same field.

The closeField message is not sent when a handler changes the field's contents using the put command.

If the lookAndFeel property is set to "Macintosh", the closeField message is generally not sent when another control (such as a button) is clicked. This is because clicked buttons do not receive the focus on Mac OS systems, and therefore the selection remains active.

The closeField message is also sent to buttons whose menuMode is "comboBox", since the type-in box in a combo box behaves like a field.

If a field is closing and its contents have not changed, the exitField message is sent instead of closeField.

closeStack

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close command, closeBackground message, closeBox property, closeCard message, closeStackRequest message, openStack message, preOpenStack message, suspendStack message

Summary

Sent to the current card when the stack closes.

Syntax

closeStack

Example Code

```
on closeStack -- automatically save changes
  save this stack
  pass closeStack
end closeStack
```

Comments

Handle the closeStack message if you want to perform cleanup or do other tasks when the user closes a window.

Comments:

A stack is closed when the user or a handler closes the stack window.

The actual closing is not triggered by the closeStack message, so trapping the message and not allowing it to pass does not prevent the stack from closing. To prevent a stack from closing, trap the closeStackRequest message.

closeStackRequest

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close command, closeStack message, openStack message, shutdownRequest message

Summary

Sent to the current card when the user tries to close a stack.

Syntax

closeStackRequest

Example Code

```
on closeStackRequest -- confirm whether to close the window
  answer "Are you sure?" with "No" or "Yes"
  if it is "Yes" then pass closeStackRequest -- allow stack to close
end closeStackRequest
```

Comments

Handle the closeStackRequest message if you want to prevent a stack from being closed.

Comments:

If the closeStackRequest handler does not pass the message or send it to a further object in the message path, the stack does not close. Passing the message allows the stack to close.

colon
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command

Summary

Equivalent to the : character (ASCII 58).

Syntax

Example Code

```
put colon into offset("/",thePath)
```

Comments

Use the colon constant as an easier-to-read replacement for ":".

colorMap

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backgroundPixel property, borderPixel property, bottomPixel property, colors property, dontDither property, focusPixel property, foregroundPixel property, hilitePixel property, lockColorMap property, privateColors property, screenColors function, screenDepth function, shadowPixel property, topPixel property, About colors and color references, Color Names Reference, Why does an image have the wrong colors?, Recipe for translating a color name to an RGB numeric triplet

Summary

Lists the colors in the current color table.

Syntax

set the colorMap to colorsList

Example Code

```
set the colorMap to myMap
if the colorMap is fixed then usePhotoIcons
```

Comments

Use the colorMap property to specify what colors Revolution uses to draw its windows when the bit depth of the screen is 8 bits or less.

Value:

The colorMap is a list of color references, one per line.

A color reference is any standard color name; or three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

The colorMap property contains the number of lines returned by the screenColors function.

Comments:

If the bit depth is greater than 8 bits, the colorMap property always reports "fixed".

The number of lines in the `colorMap` property is set when the application starts up, and does not change if you change color depths while running the application.

If you leave a line blank when setting the `colorMap`, the color corresponding to that line is left unchanged.

Cross-platform note: On Windows systems, colors 1 through 10 and 246 through 256 cannot be changed.

On Unix systems, setting the `colorMap` property sets the `privateColors` property to true.

colorNames

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

colors property, commandNames function, functionNames function, is a operator, propertyNames function, About colors and color references, Color Names Reference, Recipe for translating a color name to an RGB numeric triplet

Summary

Returns a list of the color names you can use in Transcript.

Syntax

the colorNames

colorNames()

Example Code

```
the colorNames
if field "Color" is not among the lines of the colorNames then beep
```

Comments

Use the colorNames property to check whether a particular color name can be used to specify properties such as foregroundColor and backgroundColor.

Value:

The colorNames function returns a list of color names, one per line.

Comments:

You can use the color names with the foregroundColor, backgroundColor, focusColor, topColor, bottomColor, hiliteColor, borderColor, shadowColor, selectionHandleColor, accentColor, backdrop, brushColor, and penColor properties.

colorPalette

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundColor property, colors property, foregroundColor property

Summary

Reserved for internal use.

Syntax

Example Code

Comments

colors

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundColor property, borderColor property, bottomColor property, colorNames function, effective keyword, focusColor property, foregroundColor property, hiliteColor property, patterns property, shadowColor property, topColor property, About colors and color references, Color Names Reference, Why don't buttons respect my color settings?, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies all the colors of an object, in shorthand form.

Syntax

set the colors of object to colorsList

Example Code

```
put the colors of this stack into field "Colors"  
set the colors of last button to field "Colors"  
set the colors of this stack to the colors of stack "Home"
```

Comments

Use the colors property to get all eight basic color properties at once, or to set the colors of one object to be the same as the colors of another object.

Value:

The colors of an object is a list of color references, one per line.

A color reference is any standard color name; or three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

The colors of an image contains as many lines as there are colors used in the image. The colors of all other objects contains eight lines, some of which may be empty.

Comments:

For objects other than images, you can set all these colors individually; the colors property simply provides a shorter method of dealing with all of them at once. Each line of the colors corresponds to one of the following color properties:

- Line 1: the foregroundColor
- Line 2: the backgroundColor
- Line 3: the hiliteColor
- Line 4: the borderColor
- Line 5: the topColor
- Line 6: the bottomColor
- Line 7: the shadowColor
- Line 8: the focusColor

If you leave a line blank when setting the colors, the color property corresponding to that line is left unchanged.

If the colors property of an object reports a blank line, that color is not set for the individual object, but is inherited from the object's owner. Use the form the effective colors of object to obtain the colors used for the object, whether set for the object or inherited.

If a pattern is set for an object, that pattern is used instead of the corresponding color for that object.

colorWorld

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

colorMap property, screenColors function, screenDepth function

Summary

Reports whether the screen is set to use color.

Syntax

get the colorWorld

Example Code

```
if the colorWorld then showColorIcons
```

Comments

Use the colorWorld property to determine whether the screen is color or black-and-white.

Value:

The colorWorld is true or false.

The colorWorld property is read-only and cannot be set.

Comments:

If the colorWorld property reports true, the screen is using color or grayscale. If the colorWorld is false, the screen is black-and-white.

If the system has more than one monitor, the colorWorld property reports the color status of the main screen.

The value returned by the colorWorld function is updated only when you start up the application. If you change the screen settings after starting up the application, you must quit and restart to update the colorWorld.

columnDelimiter

property

Synonyms

columnDel

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

Summary

The columnDelimiter property is not implemented and is reserved.

Syntax

Example Code

Comments

COM1:
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

close file command, COM2: keyword, COM3: keyword, COM4: keyword, COM5: keyword, COM6: keyword, COM7: keyword, COM8: keyword, COM9: keyword, LPT1: keyword, modem: keyword, open file command, printer: keyword, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands, to specify the COM 1 port on Windows systems.

Syntax

Example Code

```
read from file "COM1:" until end
```

Comments

Use the COM1: keyword to communicate through the COM 1 serial port.

Comments:

To set the port parameters (such as baud rate, handshaking, and parity), set the serialControlString property before opening the port with the open file command.

To read data from the COM 1 port, use the read from file command, specifying the keyword COM1: as the file to read from.

To write data to the COM 1 port, use the write to file command.

COM2:
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems
Not supported on Mac OS X systems
Supported on Windows systems
Not supported on UNIX systems

See Also:

close file command, COM1: keyword, COM3: keyword, COM4: keyword, COM5: keyword, COM6: keyword, COM7: keyword, COM8: keyword, COM9: keyword, LPT1: keyword, modem: keyword, open file command, printer: keyword, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands, to specify the COM 2 port on Windows systems.

Syntax

Example Code

```
read from file "COM1:" until end
```

Comments

Use the COM2: keyword to communicate through the COM 2 serial port.

Comments:

To set the port parameters (such as baud rate, handshaking, and parity), set the serialControlString property before opening the port with the open file command.

To read data from the COM 2 port, use the read from file command, specifying the keyword COM2: as the file to read from.

To write data to the COM 2 port, use the write to file command.

COM3:
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems
Not supported on Mac OS X systems
Supported on Windows systems
Not supported on UNIX systems

See Also:

close file command, COM1: keyword, COM2: keyword, COM4: keyword, COM5: keyword, COM6: keyword, COM7: keyword, COM8: keyword, COM9: keyword, modem: keyword, open file command, printer: keyword, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands, to specify the COM 3 port on Windows systems.

Syntax

Example Code

```
read from file "COM1:" until end
```

Comments

Use the COM3: keyword to communicate through the COM 3 serial port.

Comments:

To set the port parameters (such as baud rate, handshaking, and parity), set the serialControlString property before opening the port with the open file command.

To read data from the COM 3 port, use the read from file command, specifying the keyword COM3: as the file to read from.

To write data to the COM 3 port, use the write to file command.

COM4:
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems
Not supported on Mac OS X systems
Supported on Windows systems
Not supported on UNIX systems

See Also:

close file command, COM1: keyword, COM2: keyword, COM3: keyword, COM5: keyword, COM6: keyword, COM7: keyword, COM8: keyword, COM9: keyword, modem: keyword, open file command, printer: keyword, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands, to specify the COM 4 port on Windows systems.

Syntax

Example Code

```
read from file "COM1:" until end
```

Comments

Use the COM4: keyword to communicate through the COM 4 serial port.

Comments:

To set the port parameters (such as baud rate, handshaking, and parity), set the serialControlString property before opening the port with the open file command.

To read data from the COM 4 port, use the read from file command, specifying the keyword COM4: as the file to read from.

To write data to the COM 4 port, use the write to file command.

COM5:
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

close file command, COM1: keyword, COM2: keyword, COM3: keyword, COM4: keyword, COM6: keyword, COM7: keyword, COM8: keyword, COM9: keyword, modem: keyword, open file command, printer: keyword, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands, to specify the COM 5 port on Windows systems.

Syntax

Example Code

```
read from file "COM1:" until end
```

Comments

Use the COM5: keyword to communicate through the COM 5 serial port.

Comments:

To set the port parameters (such as baud rate, handshaking, and parity), set the serialControlString property before opening the port with the open file command.

To read data from the COM 5 port, use the read from file command, specifying the keyword COM5: as the file to read from.

To write data to the COM 5 port, use the write to file command.

COM6:
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems
Not supported on Mac OS X systems
Supported on Windows systems
Not supported on UNIX systems

See Also:

close file command, COM1: keyword, COM2: keyword, COM3: keyword, COM4: keyword, COM5: keyword, COM7: keyword, COM8: keyword, COM9: keyword, modem: keyword, open file command, printer: keyword, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands, to specify the COM 6 port on Windows systems.

Syntax

Example Code

```
read from file "COM1:" until end
```

Comments

Use the COM6: keyword to communicate through the COM 6 serial port.

Comments:

To set the port parameters (such as baud rate, handshaking, and parity), set the serialControlString property before opening the port with the open file command.

To read data from the COM 6 port, use the read from file command, specifying the keyword COM6: as the file to read from.

To write data to the COM 6 port, use the write to file command.

COM7:
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems
Not supported on Mac OS X systems
Supported on Windows systems
Not supported on UNIX systems

See Also:

close file command, COM1: keyword, COM2: keyword, COM3: keyword, COM4: keyword, COM5: keyword, COM6: keyword, COM8: keyword, COM9: keyword, modem: keyword, open file command, printer: keyword, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands, to specify the COM 7 port on Windows systems.

Syntax

Example Code

```
read from file "COM1:" until end
```

Comments

Use the COM7: keyword to communicate through the COM 7 serial port.

Comments:

To set the port parameters (such as baud rate, handshaking, and parity), set the serialControlString property before opening the port with the open file command.

To read data from the COM 7 port, use the read from file command, specifying the keyword COM7: as the file to read from.

To write data to the COM 7 port, use the write to file command.

COM8:
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

close file command, COM1: keyword, COM2: keyword, COM3: keyword, COM4: keyword, COM5: keyword, COM6: keyword, COM7: keyword, COM9: keyword, modem: keyword, open file command, printer: keyword, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands, to specify the COM 8 port on Windows systems.

Syntax

Example Code

```
read from file "COM1:" until end
```

Comments

Use the COM8: keyword to communicate through the COM 8 serial port.

Comments:

To set the port parameters (such as baud rate, handshaking, and parity), set the serialControlString property before opening the port with the open file command.

To read data from the COM 8 port, use the read from file command, specifying the keyword COM8: as the file to read from.

To write data to the COM 8 port, use the write to file command.

COM9:
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

close file command, COM1: keyword, COM2: keyword, COM3: keyword, COM4: keyword, COM5: keyword, COM6: keyword, COM7: keyword, COM8: keyword, modem: keyword, open file command, printer: keyword, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands, to specify the COM 9 port on Windows systems.

Syntax

Example Code

```
read from file "COM1:" until end
```

Comments

Use the COM9: keyword to communicate through the COM 9 serial port.

Comments:

To set the port parameters (such as baud rate, handshaking, and parity), set the serialControlString property before opening the port with the open file command.

To read data from the COM 9 port, use the read from file command, specifying the keyword COM9: as the file to read from.

To write data to the COM 9 port, use the write to file command.

combine

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

& operator, [] keyword, split command, How to find out whether a container is an array

Summary

Transforms an array into a list.

Syntax

combine array {using | by | with} primaryDelimiter [and secondaryDelimiter]

Example Code

```
combine myArray using comma  
combine monthlyReceivables using return and ";"
```

Comments

Use the combine command to display an array in a field or to process an array using string operators, functions, and chunk expressions.

Parameters:

The arrayName is an array variable.

The primaryDelimiter is a character whose ASCII value is in the range 1 to 127, or an expression that evaluates to such a character.

The secondaryDelimiter is a character (other than the primaryDelimiter) whose ASCII value is in the range 1 to 127, or an expression that evaluates to such a character.

Comments:

The combine command combines the elements of the array into a single variable. After the command is finished executing, the variable specified by array is no longer an array.

The elements of the original array are separated by the primaryDelimiter. For example, if the primaryDelimiter is return, the content of each element of the original array appears on a separate line.

If you specify a secondaryDelimiter, the key corresponding to each element is added to the element's content, separated from the content by the secondaryDelimiter. For example, if the primaryDelimiter is return and the secondaryDelimiter is tab, each line of the resulting variable contains an element's key, a tab character, and the element's content. If you don't specify a secondaryDelimiter, then the keys are lost in the transformation.

Note: The order of the elements is not alphabetical or chronological; it is based on the internal hash order of the array. To alphabetize the list, use the sort command:

```
combine monthlyReceivables using return and comma  
sort lines of monthlyReceivables by item 2 of each
```

comboBox

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cascade keyword, menuMode property, option keyword, popup keyword, pulldown keyword, style property, tabbed keyword, Object menu > New Control > Combo Box Menu

Summary

Specifies one of the menu types that can be used with the menuMode property.

Syntax

Example Code

```
set the menuMode of button "Aptness" to comboBox
```

Comments

Use the comboBox keyword to create a combo box menu.

Comments:

Combo box menus are normally used to designate a state. Choosing an item from a combo box changes the current setting.

Using the text box at the top of the combo box, the user can specify a state that is not in the menu's list of states. This menu type is suitable for situations in which the user probably will choose one of the predefined states, but might need to specify a different state.

Note: If a button's menuMode is set to "comboBox", the button receives field messages. For example, when the user clicks in the editable field, an openField message is sent to the button.

comma

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

, operator, constant command, How to include a comma in a parameter

Summary

Equivalent to the , character (ASCII 44).

Syntax

Example Code

```
put "3" & comma & "287" into dollarAmount  
if char 3 of it is comma then get field "Circus"
```

Comments

Use the comma constant as an easier-to-read replacement for ",".

commandChar

property

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

acceleratorKey property, markChar property, menuItem keyword, mnemonic property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the commandChar of menuItem to character

Example Code

Comments

In HyperCard, the commandChar property determines the Command key combination for a menu item.

A handler can set the commandChar to any value without causing a script error, but the actual Command key combination is not changed.

commandKey

function

Synonyms

cmdKey

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

altKey function, commandKeyDown message, controlKey function, down constant, keyDown message, keysDown function, keyUp message, metaKey function, optionKey function, shiftKey function, up constant

Summary

Returns the state of the Command key.

Syntax

the commandKey

commandKey()

Example Code

```
put commandKey() after keyHistory  
if the commandKey is down then edit script of me
```

Comments

Use the commandKey function to check whether the Command key is being pressed.

Value:

The commandKey function returns "down" if the key is pressed and "up" if it's not.

Comments:

On Unix and Windows systems, the commandKey function returns the same value as the controlKey function: the two functions are synonyms.

commandKeyDown

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

commandKey function, controlKeyDown message, keyDown message, keysDown function, rawKeyDown message

Summary

Sent when a Command key combination (Control-key on Unix or Windows) is pressed.

Syntax

commandKeyDown keyName

Example Code

```
on commandKeyDown theKey -- make Command-5 go back
  if theKey is "5" then go recent card else pass commandKeyDown
end commandKeyDown
```

Comments

Handle the commandKeyDown message if you want to provide Command-key or Control-key shortcuts (other than those provided in menus and button accelerators).

Parameters:

The keyName is the actual character of the pressed key.

Comments:

The commandKeyDown message is sent to the active (focused) control, or to the current card if no control is focused.

If the Command key is pressed along with the Return, Tab, Backspace, Delete, or Enter key, with an arrow key, or with a function key, no commandKeyDown message is sent. Instead, the returnKey, tabKey, backspaceKey, deleteKey, enterKey, arrowKey or functionKey message is sent. To trap a combination such as Command-Return or Control-Return, use a returnKey handler and check the commandKey function inside the handler.

Cross-platform note: On Mac OS systems, the `commandKeyDown` message is sent when the user presses a key while holding down the Command key. On Unix or Windows systems, the `commandKeyDown` message is sent when the user presses the Control key, whether or not another key is pressed at the same time.

commandNames

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constantNames function, functionNames function, on control structure, propertyNames function, variableNames function

Summary

Returns a list of all built-in commands in Transcript.

Syntax

the commandNames

commandNames()

Example Code

```
the commandNames
```

```
if field "Name" is among the lines of the commandNames then exit repeat
```

Comments

Use the commandNames function to check whether a particular command already exists in Transcript, to avoid using a reserved word for your own custom handlers.

Value:

The commandNames function returns a list of command names, one per line.

Comments:

The commandNames function returns all the commands that can be used in Transcript, including synonyms.

compact

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

delete command, save command

Summary

Reclaims unused space within a stack's file structure.

Syntax

compact stack

Example Code

```
compact this stack  
compact stack "Encyclopedia"
```

Comments

When you cut or delete a card, some free space is left within the stack. This empty space is reclaimed when you choose File menu Save. Use the compact command to reclaim this empty space without having to save the stack.

Parameters:

The stack is any stack reference.

Comments:

This command is especially useful when a handler performs multiple rearrangements of cards without saving the stack.

The "Save" menu item in the File menu automatically compacts the stack. However, the save command does not do so.

compound function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

annuity function, round function

Summary

Returns the principal plus accrued interest of an investment.

Syntax

`compound(interestPerPeriod,numberOfPeriods)`

Example Code

```
compound(.05,20)  
put theInvestment * compound(yearlyInterest,numberYears) into endValue
```

Comments

Use the compound function to calculate the value of an investment that earns interest.

Parameters:

The `interestRate` is a positive number. The `interestRate` is expressed as a fraction of 1 so, for example, an 8% rate is written .08.

The `numberOfPeriods` is a positive number.

Value:

The compound function returns a positive number.

Comments:

The formula for the value of an compound-interest-bearing account is

$$(1 + \text{interestRate})^{(\text{numberOfPeriods})}$$

The compound function calculates this value.

The numberOfPeriods and the interestRate must use the same unit of time. For example, if the periods are months, the interest rate is the interest per month.

compress

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

base64Encode function, decompress function, MD5Digest function, URLEncode function

Summary

Returns a gzip-compressed string.

Syntax

the compress of data
`compress(data)`

Example Code

```
compress(URL "file:image.pict")  
put compress(field "Outgoing") into URL "binfile:data.gz"
```

Comments

Use the compress function to compress data to a smaller size for transmission.

Parameters:

The data is a string of binary data of any length.

Value:

The compress function returns a string of binary data.

Comments:

The compress function is the inverse of the decompress function.

The compressed result is typically about half to a third the size of the original data, although different results may be obtained depending on the amount of data and whether it has already been compressed.

Important! The value returned by the compress function consists of binary data and may include control characters, so displaying it on screen or trying to edit it may produce unexpected results. If you use a URL to place the returned data in a file, it's important to use the binfile URL scheme; using the file URL scheme will corrupt binary data.

For technical information about the format used by the compress function, see RFC 1952 at [<http://www.ietf.org/rfc/rfc1952.txt>](http://www.ietf.org/rfc/rfc1952.txt). The compress function uses the zlib compression library.

condensed

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

expanded keyword, plain keyword, textStyle property

Summary

Used with the textStyle property to indicate horizontally compressed text.

Syntax

Example Code

```
set the textStyle of line 1 of field "Main" to condensed
if the textStyle of the mouseControl is "condensed" then beep
```

Comments

Use the condensed keyword to reduce the width of text without changing its size.

Comments:

You can condense an object (which condenses all the text displayed by the object) or a chunk in a field or button.

To condense text without removing other styles (such as bold or italic), add the condensed keyword to the end of the textStyle. The following example condenses the text of a field without affecting its existing style setting:

```
if the textStyle of field "Dark" is empty then
    set the textStyle of field "Dark" to condensed
else
    set the textStyle of field "Dark" to
        (the textStyle of field "Dark") & comma & "condensed"
end if
```

constant

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constantNames function, global command, local command, variableNames function, About containers, variables, and sources of value, Recipe for 99 Bottles of Beer on the Wall (using send), Recipe for a "roll credits" effect

Summary

Declares one or more labels to assign to constant values.

Syntax

constant constantsList

Example Code

```
constant defaultName="Jones"  
constant entryA="EF9993333WX786",entryB="GJ773281YX342"
```

Comments

Use the constant command to assign an unchanging value to a keyword.

Parameters:

The constantsList consists of one or more name=value pairs, separated by commas:

- ï The name is any string.
- ï The value is any literal string.

Comments:

Constants can be numbers, characters, logical values, or strings.

A constant cannot be defined as an array, only as a single value.

Note: Choose easy-to-understand names for constants to improve the readability of your code. Use constants as substitutes for long or convoluted strings.

If you place the constant statement in a handler, you can use the constant anywhere in the handler. If you place the constant statement in a script outside any handler, you can use the constant anywhere in the handlers of that script.

Once you have defined a constant, you cannot redefine it in the same handler; doing so causes an execution error.

You can use the constant command to redefine Transcript's built-in constants within a script or handler, but doing so makes your code harder to read and maintain. To find out whether a word is already defined as a built-in constant, use the constantNames function.

Tip: To see a list of built-in constants, open the Documentation window, click Transcript Dictionary, and choose "Constants" from the menu at the top of the window.

constantMask

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

alwaysBuffer property, filename property, imagePixmapID property, Why do some frames of an animated GIF look strange?

Summary

Specifies whether animated GIFs ignore changes in the mask data when displaying the GIF.

Syntax

set the constantMask of image to {true | false}

Example Code

```
set the constantMask of image "Demo" to true
```

Comments

Use the constantMask property to enable display of certain animated GIF images.

Value:

The constantMask of an image is true or false.

By default, the constantMask property of newly created images is set to false.

Comments:

Some animated GIF images use an optimization technique in which the mask data is used to hold information about differences between successive frames. When played, these GIFs can have display problems because the mask data is being used in a non-standard way. Set the constantMask to true to correct the problem with these GIFs.

Setting the constantMask to true may have some undesirable side effectsóthe image's appearance may be incorrect if you move it while the animation is playing, or if there is an object underneath itóso use this property only when necessary.

Note: If the constantMask is set to true, clicking a transparent portion of a frame may unexpectedly send messages such as mouseUp to the image, instead of the object underneath the image. This is

because when deciding whether the user has clicked a transparent pixel, Revolution uses the masks for all the frames, rather than for the current frame, if the `constantMask` is true.

Important! Setting an image's `constantMask` property to true can cause problems with the image's appearance if you move it under script control while the animation is playing, or if there's an object underneath it. Avoid setting the `constantMask` property to true unless necessary.

constantNames

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

commandNames function, constant command, functionNames function, propertyNames function, variableNames function

Summary

Returns a list of all built-in constants in Transcript.

Syntax

the constantNames

constantNames()

Example Code

```
the constantNames
```

```
if it is among the lines of the constantNames then askForAnotherName
```

Comments

Use the constantNames function to check whether a particular constant already exists in Transcript, to avoid using a reserved word for your own constants and variables.

Value:

The constantNames function returns a list of constant names, one per line.

Comments:

The constantNames function returns all the constants that are predefined in Transcript, including synonyms.

constraints

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

nodes property, pan property, tilt property, zoom property

Summary

Limits the pan, tilt, and zoom in a QuickTime VR movie.

Syntax

get the constraints of player

Example Code

```
get the constraints of player "Arctic"  
set the pan of me to item 2 of line 1 of the constraints of me
```

Comments

Use the constraints property to find out where the user is in a QuickTime VR movie.

Value:

The constraints is a list consisting of three lines:

- The minimum and maximum pan, separated by commas.
- The minimum and maximum tilt, separated by commas.
- The minimum and maximum zoom, separated by commas.

Each value is a number.

This property is read-only and cannot be set.

Comments:

The user can move the view of a QuickTime VR movie using the navigational controls in the player; a handler can change the view by setting the player's pan and tilt properties, and change the view angle by changing the zoom property.

The zoom, pan, and tilt are limited by the value of the player's constraints property. If you specify a value greater than the range permitted by the constraints, the property is set to the highest permitted value. If you specify a value less than the range permitted by the constraints, the property is set to the lowest permitted value.

If the player does not contain a QuickTime VR movie, its constraints property is empty.

contains

operator

Synonyms

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

= operator, is among operator, is in operator, Operator Precedence Reference

Summary

Compares two strings and evaluates to true if the first contains the second, false if not.

Syntax

string contains substring

Example Code

```
"ABC" contains "A" -- evaluates to true  
"123" contains "13" -- evaluates to false
```

Comments

Use the contains operator to find out whether a string contains a substring.

Parameters:

The string and substring are both strings of characters, or expressions that evaluate to strings.

Comments:

If the substring is found in the string, the contains operation evaluates to true.

If each character of the substring can be found in the string, but the characters are separated, the contains operation evaluates to false.

control

keyword

Synonyms
part

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
layer property, mouseControl function, number function

Summary
Designates all object types that can be placed on a card.

Syntax

Example Code

```
put the number of controls into index  
select control 5
```

Comments

Use the control keyword in an object reference.

Comments:

A control is any object that can be placed on a card: a button, field, scrollbar, image, graphic, player, EPS object, or group.

You can refer to a control by number or by name. The expression control 2 refers to the control whose layer is 2. The expression control "this" refers to the first control whose short name is "this".

controlKey

function

Synonyms
ctrlKey

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

altKey function, commandKey function, commandKeyDown message, down constant, keyDown message, keysDown function, keyUp message, metaKey function, optionKey function, shiftKey function, up constant

Summary

Returns the state of the Control key.

Syntax

the controlKey
controlKey()

Example Code

```
put (the controlKey is up) and (the shiftKey is up) into ctrlShift  
if controlKey() is down then go back
```

Comments

Use the controlKey function to check whether the user is pressing the Control key.

Value:

The controlKey function returns down if the key is pressed and up if it's not.

On Unix and Windows systems, the commandKey function returns the same value as the controlKey function: the two functions are synonyms.

controlKeyDown

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

commandKeyDown message, keyDown message, keysDown function, rawKeyDown message

Summary

Sent when a Control key combination is pressed.

Syntax

controlKeyDown keyName

Example Code

```
on controlKeyDown myKey -- make Control-D delete the current card
  if theKey is "D" then delete this card
  else pass controlKeyDown
end controlKeyDown
```

Comments

Handle the controlKeyDown message if you want to provide Control-key shortcuts on Mac OS systems.

Parameters:

The keyName is the actual character of the pressed key.

Comments:

The message is sent to the active (focused) control, or to the current card if no control is focused.

If the Control key is pressed along with the Return key, Tab key, or Enter key, or with an arrow key, no controlKeyDown message is sent. Instead, the returnKey, tabKey, enterKey, or arrowKey message is sent. To trap a combination such as Control-Return, use a returnKey handler and check the controlKey function inside the handler.

Cross-platform note: On Mac OS systems, the controlKeyDown message is sent when a control key combination is pressed. On Windows systems, a control key combination sends a commandKeyDown message instead, and the controlKeyDown message is never sent. On Unix systems, the controlKeyDown message is sent only if Mod2 has been defined.

convert

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

abbreviated keyword, centuryCutoff property, date function, dateItems keyword, english keyword, internet keyword, is a operator, long keyword, seconds function, seconds keyword, short keyword, system keyword, time function, twelveHourTime property

Summary

Changes a date, a time, or a date and time to a specified format.

Syntax

convert dateAndTime [from format [and format]] to format [and format]

Example Code

```
convert "11/22/90" to long english date
convert it from internet date to system date
convert lastChangedTime to abbreviated time and long date
convert the date && the time to seconds
```

Comments

Use the convert command to change a date or time to a format that's more convenient for calculation or display.

Parameters:

The dateAndTime is a string or container with a date, a time, or a date and time separated by a space, tab, or return character.

The format is one of the following (examples are February 17, 2000 at 10:13:21 PM, in the Eastern time zone, using US date and time formats):

- short date: 2/17/00
- abbreviated date: Thu, Feb 17, 2000
- long date: Thursday, February 17, 2000
- short time: 10:13 PM
- abbreviated time: 10:13 PM
- long time: 10:13:21 PM

- internet date: Thu, 17 Feb 2000 22:13:21 -0500
- seconds: the number of seconds since the start of the eon
- dateItems: 2000,2,17,22,13,21,5

If you specify both a date and time format, they can be in either order and must be separated by a space. The resulting date and time are in the order you provided, separated by a space. If you specify seconds or dateItems, you can request only one format.

Comments:

If the dateAndTime is a container, the converted date and time is placed in the container, replacing the previous contents. If the dateAndTime is a string, the converted date and time is placed in the it variable.

The dateItems format is a comma-separated list of numbers:

- the year
- the month number
- the day of the month
- the hour in 24-hour time
- the minute
- the second
- the numeric day of the week where Sunday is day 1, Monday is day 2, and so forth

The convert command can handle dates in dateItems format where one or more of the items is out of the normal range. This means you can add arbitrary numbers to an item in the dateItems and let the convert command handle the calculations that span minute, hour, day, month, and year boundaries.

For example, suppose you start with 9:30 AM , convert that time to dateItems format, then add 45 minutes to item 5 (the minute) of the resulting value. This gives you 75 as the minute. When you convert the value to any other time format, the convert command automatically converts "75 minutes" to "1 hour and 15 minutes":

```
convert "9:30 AM" to dateItems
add 45 to item 5 of it
convert it to time -- yields "10:15 AM"
```

You can optionally use the english or system keyword before the short, abbreviated, or long date or time. If the useSystemDate is true, or if you use the system keyword, the user's current system preferences are used to format the date or time. Otherwise, the standard US date and time formats are used.

The internet date, seconds, and dateItems formats are invariant and do not change according to the user's preferences.

Note: If you convert a date without a time to a form that includes the time, the time will be given as 2 AM. (This is the default time because it is the time of day when Daylight Savings Time begins and ends.)

Note: The range of dates that the convert command can handle is limited by the operating system's date routines. In particular, Windows systems are limited to dates after 1/1/1970.

Changes to Transcript:

The ability to use the date and time format preferred by the user was introduced in version 1.1. In previous versions, the convert command, along with the time and date functions, consistently used the standard U.S. format, even if the operating system's settings specified another language or date and time format.

The ability to specify a format to convert from was introduced in version 1.1. In previous versions, Revolution automatically guessed the format to convert from.

convertOctals

property

Synonyms

Objects

local

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

baseConvert function

Summary

Specifies whether numbers with a leading zero are assumed to be octal numbers.

Syntax

set the convertOctals to {true | false}

Example Code

```
set the convertOctals to true
```

Comments

Use the convertOctals property to make it easier to work with octal numbers.

Value:

The convertOctals is true or false.

By default, the convertOctals property is set to false.

Comments:

If the convertOctals property is set to true, numbers that start with a zero are treated as octal (base 8) numbers, instead of decimal (base 10). If the convertOctals is false, all numbers (except hexadecimal numbers, which start with 0x) are treated as decimal.

Since the convertOctals is a local property, its value is reset to false when the current handler finishes executing. It retains its value only for the current handler, and setting it in one handler does not affect its value in other handlers it calls.

copy

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clipboardData property, clone command, copyResource function, cut command, paste command, How to copy a card, How to copy an image to the clipboard, How to duplicate an object, How to move cards from one stack to another, Edit menu > Copy

Summary

Copies selected text or an object to the clipboard or to a card, group, or stack.

Syntax

copy [object [to {card | group | stack}]] | [chunk of field]

Example Code

```
copy -- copies selected object or text to clipboard
copy button "Hello" -- copies the button to the clipboard
copy this card to stack "Clippings" -- leaves clipboard unchanged
copy word -3 to -1 of field "Help" -- copies last three words
```

Comments

Use the copy command to place objects or text on the clipboard, or to make a copy of an object without changing the contents of the clipboard.

Parameters:

The object is any available object. If no object or chunk is specified, the copy command copies whatever is currently selected.

If a card, group, or stack is specified, the object is placed there instead of being placed in the clipboard, and the clipboard is left unchanged.

Comments:

If a card is specified, the copied object must be a control or group.

If a group is specified, the copied object must be a control.

If a stack is specified, the copied object must be a card.

If a chunk of a field is specified, the specified text is copied to the clipboard (including text styles).

If a destination card, group, or stack is specified, the copy command places the ID property of the newly created object in the it variable, and the appropriate message is sent.

Note: When copying from a list field, only entire lines can be copied. If you specify a chunk that's smaller than a line or that crosses line boundaries, the copy command copies the entire line or lines containing the chunk.

Changes to Transcript:

The ability to copy a chunk of a field directly (without first selecting it) was added in version 2.0.

copyKey

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

copy command, cutKey message, deleteKey message, pasteKey message

Summary

Sent when the user presses the key combination equivalent to the Copy menu item.

Syntax

copyKey

Example Code

```
on copyKey
  if the selection is empty then beep -- nothing to copy
  pass copyKey
end copyKey
```

Comments

Handle the copyKey message if you want to change the normal copying process, or prevent use of the Copy key combination without changing the menu.

Comments:

The Revolution development environment traps the copyKey message, unless "Suspend Revolution UI" is turned on in the Development menu. This means that the copyKey message is not received by a stack if it's running in the development environment.

The copyKey message is sent when the user presses Command-C (on Mac OS systems), Control-C (on Windows systems), Control-Insert (on Unix systems), or the keyboard Copy key.

The message is sent to the active (focused) control, or to the current card if no control is focused.

copyResource

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

copy command, deleteResource function, getResources function, resfile keyword, sysError function

Summary

Copies a resource from one Mac OS file to another.

Syntax

copyResource(file,destinationFile,resType,{resID | resName}[,newID])

Example Code

```
get copyResource("Template","New Build","vers",1,1)
put copyResource(theFile,anotherFile,"XCMD","SetPort") into trashVar
get copyResource(field "Source",currentFile,nextResource,200)
```

Comments

Use the copyResource function to copy resources (such as icons, externals, and version resources) from one file to another.

Parameters:

The file is the name and location of the file containing the resource. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

The destinationFile is the name and location of the file you want to copy the resource to.

The resType is the 4-character type of the resource you want to move.

The resID is an integer that specifies the resource ID of the resource you want to copy.

The resName is the name of the resource you want to copy.

The newID is an integer that specifies the resource ID for the newly-copied resource.

Value:

The copyResource function always returns empty.

Comments:

If the file does not exist, the result is set to "can't find file".

If the destinationFile does not exist, the copyResource function creates the file, but the result is set to "can't create resource fork for destination file". If the destinationFile exists but has no resource fork, the copyResource function creates the resource fork and copies the resource to it.

If the destinationFile is open, the result is set to "Can't open resource fork".

If you don't specify a newID, the new resource has the same ID as the original resource. Specifying a newID does not change the resource ID of the original resource in the file; it only affects the copy in the destinationFile.

COS

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

acos function, pi constant, sin function, tan function

Summary

Returns the cosine of an angle (in radians).

Syntax

the cos of angle
`cos(angle)`

Example Code

```
cos(pi) -- returns -1  
the cos of 0 -- returns 1  
put cos(the startAngle of graphic 3) into field "Cosine"
```

Comments

Use the cos function to find the cosine of an angle.

Parameters:

The angle is a positive or negative number, or an expression that evaluates to a number.

Value:

The cos function returns a number between -1 and 1.

Comments:

The cos function repeats every 2π radians: $\cos(x) = \cos(x + 2 * \pi)$, for any value of x.

The cos function requires its input to be in radians. To provide an angle in degrees, use the following custom function:

```
function cosInDegrees angleInDegrees  
  return cos(angleInDegrees * pi / 180)  
end cosInDegrees
```

create

command

Synonyms
new

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clone command, create card command, create stack command, newButton message, newEPS message, newField message, newGraphic message, newImage message, newPlayer message, How to add an object to a group, How to create a card template for a stack, Why do objects look different when created in a standalone?, Why is a control the wrong size when created by a handler?, File menu > Import as Control, File menu > New Referenced Control, Object menu > New Control

Summary

Creates a new object on the current card.

Syntax

create [invisible] type [name] [in group]

Example Code

```
create button "Click Me"  
create invisible field in first group
```

Comments

Use the create command to make a new control or grouped control.

Parameters:

The type is any control that can be on a card: field, button, image, scrollbar, graphic, player, or EPS.

The name is the name of the newly created object. If you don't specify a name, the object is created with a default name.

The group is any group that's on the current card. If you specify a group, the new object is a member of the group, and exists on each card that has the group. If you don't specify a group, the object is created on the current card and appears only on that card.

Comments:

The new object takes its properties from the corresponding template; for example, newly created buttons match the properties of the templateButton.

If you use the create invisible form, the object is created with its visible property set to false, so it cannot be seen. Use this form to create a hidden object, change its appearance or position, then make it visible.

The create command places the ID property of the newly created object in the it variable.

When the new control is created, the Pointer tool is automatically chosen. If you use the create command in a handler, you can use the following statement after the create command to resume using the Browse tool:

```
send "choose browse tool" to me in 1 tick
```

Note: In the development environment, after an object is created, Revolution automatically resets the corresponding template to its default values. This means that if you change an object template and then create several objects of that type, only the first object will reflect your settings. To prevent Revolution from automatically setting the template back to its defaults, set the lockMessages property to true before creating the objects:

```
set the borderWidth of the templateButton to 8
lock messages
repeat for 5 times
  create button
end repeat
unlock messages
```

Revolution resets the template only when in the development environment, not in standalone applications.

Tip: To create a control in a specific stack, first set the defaultStack to the stack where you want to create the new control:

```
set the defaultStack to "My Stack"
create button "My Button"
```

create alias

command

Synonyms

new alias

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

aliasReference function, create folder command, defaultFolder property, rename command, About filename specifications and file paths, How to make an alias, symbolic link, or shortcut to a file

Summary

Makes a new alias, symbolic link, or shortcut to the specified file.

Syntax

create alias aliasPath to file originalFilePath

Example Code

```
create alias "Music Shortcut" to file "Music"  
create alias preferredFileLocation to file "../Startup"
```

Comments

Use the create alias command to create a second icon for a file, in a location that is handier for the user.

Parameters:

The aliasPath specifies the name and location of the alias to be created. If you specify a name but not a location, the alias is created in the defaultFolder.

The originalFilePath is the name and location of the file you want to make an alias to. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

Comments:

You cannot create an alias to a disk volume or a folderóonly to a file.

To find out which file an alias refers to, use the aliasReference function.

Important! When creating an alias on Windows systems, make sure the alias's name ends with the extension ".lnk" to ensure that it behaves properly when the user double-clicks it.

create card

command

Synonyms

new card

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clone command, create command, create stack command, newCard message, templateCard keyword, Recipe for a Cards menu, Why does the window change size when I create a card?, Why do objects look different when created in a standalone?, Why do unwanted objects appear on new cards?, Object menu > New Card

Summary

Makes a new card with the same background(s) as the current card.

Syntax

create card [cardName]

Example Code

```
create card
card card "Employees"
create card (field "Name" of card "Template")
```

Comments

Use the create card command to add a new card to the current stack.

Parameters:

The cardName is the name property of the new card. If you don't specify a cardName, the new card's name is empty.

Comments:

The new card is placed after the current card in the defaultStack. When the card is created, you go to the new card.

Any groups on the current card whose backgroundBehavior is set to true will be placed on the newly-created card automatically.

The create card command places the ID property of the newly created card in the it variable.

Tip: To create a card in a specific stack, first set the defaultStack to the stack where you want to create the new card:

```
set the defaultStack to "My Stack"  
create card
```

create folder

command

Synonyms

create directory, new folder, new directory

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

defaultFolder property, folders function, open file command, rename command, revCopyFolder command, revDeleteFolder command, revMoveFolder command, save command, specialFolderPath function, sysError function, tempName function, About filename specifications and file paths

Summary

Makes a new folder.

Syntax

create folder pathName

Example Code

```
create folder "Translated Documents"  
create folder "../My Files/Temp Files"
```

Comments

Use the create folder command to make a new folder for a stack's use. For example, if the stack generates temporary text files, creating a folder for the files prevents them from cluttering the user's system.

Parameters:

The pathName specifies the name and location of the folder. If you specify a name but not a location, the folder is created in the defaultFolder.

Comments:

To create a file, use the open file command.

create stack

command

Synonyms
new stack

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clone command, create command, newStack message, save command, templateStack keyword, About main stacks, substacks, and the organization of a stack file, Why do objects look different when created in a standalone?, File menu > New Mainstack, File menu > New Substack of (main stack name)

Summary

Makes a new blank stack.

Syntax

create [invisible] stack [stackName [with background]]

Example Code

```
create stack "Test"  
create stack (field 3) with background "Standard Navigation"
```

Comments

Use the create stack command to make a new stack.

Parameters:

The stackName is the name of the newly-created stack. If you don't specify a stackName, the new stack's name is "Untitled" and a number.

The background is a reference to any group in the defaultStack. This group, if specified, is included in the new stack.

Comments:

The newly created stack is not saved to a file until you explicitly save it. If you close it without saving it, the stack disappears.

If you use the create invisible stack form, the stack is created with its visible property set to false, so it cannot be seen. Use this form to create a hidden window, change its appearance or position, then make it visible.

The create stack form of this command is equivalent to choosing File menu New Mainstack.

CRLF

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command, read from file command, return constant, write to file command

Summary

Equivalent to a carriage return (ASCII 13, Control-M) followed by a line feed (ASCII 10, Control-J).

Syntax

Example Code

```
read from file myData until CRLF
put line x of field "Information" & CRLF after dataToExport
```

Comments

Use the CRLF constant as an easier-to-read substitute for numToChar(13) & numToChar(10).

Comments:

The CRLF constant is needed because you can't type the characters it represents in a script.

The line feed character is the standard end-of-line delimiter on Unix systems. The end-of-line delimiter for Mac OS systems is a carriage return, and the end-of-line delimiter for Windows systems is a carriage return followed by a line feed.

crop

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

flip command, imageData property, rectangle property, rotate command

Summary

Trims an image.

Syntax

crop image to left,top,right,bottom

Example Code

```
crop image "photo" to 100,100,200,200  
crop image myImage to the rect of graphic "Overlay"
```

Comments

Use the crop command to remove parts of an image, or to extend the image's edges.

Parameters:

The image is any image reference.

The left, top, right, and bottom describe the new rectangle of the image.

Comments:

The crop command changes the image's rectangle property to the new rectangle. Unlike changing the rectangle directly, the crop command removes image data outside the new rectangle, instead of scaling the image to fit.

If the crop rectangle is smaller than the image's original rectangle, the parts of the image outside the crop rectangle are removed. This is a permanent operation; the cropped portions do not reappear if you use the crop command to change the image's rectangle back to the original size.

If the crop rectangle is larger than the image's original rectangle, blank pixels are added to one or more edges of the image.

Important! The crop command cannot be used on a referenced image. Doing so will cause an execution error.

CROSS
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, cursor property, plus constant, seven constant

Summary

Equivalent to the number 7.

Syntax

Example Code

```
set the cursor to cross
```

Comments

Use the cross constant to set the cursor to a crosshairs shape, suitable for selecting a rectangular area.

Comments:

The following two statements are equivalent:

```
set the cursor to cross  
set the cursor to 7
```

However, the first is easier to read and understand in Transcript code.

Important! If you use the cross cursor or other standard cursors in your application, you must include it when you create your standalone. Make sure the "Cursors" option on the Inclusions tab in Step 3 of the Distribution Builder is checked.

currentFrame

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentTime property, frameCount property, palindromeFrames property, repeatCount property, Why do some frames of an animated GIF look strange?

Summary

Specifies the current frame of an animated GIF.

Syntax

set the currentFrame of image to frameNumber

Example Code

```
set the currentFrame of image "Process" to 1 -- back to the beginning
```

Comments

Use the currentFrame property to check where an animated GIF image is in its sequence. Set this property in order to change the current frame of an animated GIF: for example, to examine the animation frame by frame.

Value:

The currentFrame of an image is a positive integer.

By default, the currentFrame property of newly created images is set to 1.

Comments:

If the image is not a GIF, the currentFrame property always reports 1, and setting it has no effect.

currentNode

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

nodeChanged message, nodes property

Summary

Specifies the current node in a QuickTime VR movie.

Syntax

set the currentNode of player to nodeID

Example Code

```
set the currentNode of player "Arctic" to 3  
put the currentNode of player myPlayerName into myNode
```

Comments

Use the currentNode property to find out where the user is in a QuickTime VR movie.

Value:

The currentNode is a positive integer.

Comments:

Each node of a QuickTime VR movie is a viewpoint. The movie author sets the nodes during development of the movie. The user can change nodes using the navigational controls in the player; a handler can change nodes by setting the player's currentNode property.

If the player does not contain a QuickTime VR movie, its currentNode property is zero.

currentPage

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

pageCount property, postScript property, prolog property

Summary

Specifies the page being displayed in a multi-page EPS object.

Syntax

set the currentPage of EPS to pageNumber

Example Code

```
set the currentPage of EPS 6 to (1 + the currentPage of EPS 6)
```

Comments

Use the currentPage property to switch between the pages of an EPS object.

Value:

The currentPage of an EPS object is an integer between 1 and the EPS object's pageCount.

Comments:

This property is supported only on Unix systems with Display PostScript installed.

currentTime

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentTimeChanged message, duration property, endTime property, play command, playSelection property, start command, startTime property, stop command, timeScale property

Summary

Specifies the elapsed time in a movie or sound.

Syntax

set the currentTime of player to number

Example Code

```
set the currentTime of player "Open Door" to 300
```

Comments

Use the currentTime property during playing to find out the progress of the movie or sound, or to skip ahead or back.

Value:

The currentTime of a player is an integer between zero and the player's duration.

Comments:

The currentTime is the number of the interval where the current position is.

The number of intervals per second is specified by the player's timeScale property. The total number of intervals is given in the player's duration property.

currentTimeChanged

message

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentTime property, play command, playPaused message, selectionChanged message, showController property

Summary

Sent to a player when the user switches to another frame.

Syntax

currentTimeChanged newTime

Example Code

```
on currentTimeChanged theInterval -- display the time in a field
  put theInterval into field "Time Code"
end currentTimeChanged
```

Comments

Handle the currentTimeChanged message if you want to do something when the user changes the current time in a player.

Parameters:

The newTime is the number of the interval where the player is after the change.

Comments:

The user can change the current time by clicking in the controller bar or by using the arrow keys when the player is the active (focused) control.

The number of intervals per second is specified by the player's timeScale property. The total number of intervals is given in the player's duration property.

The actual process of changing the player's current time is not triggered by the currentTimeChanged message, so trapping the message and not allowing it to pass does not prevent the user from changing the player's current time manually.

Changes to Transcript:

The newTime parameter was introduced in version 2.0. In previous versions, the currentTimeChanged message did not have a parameter.

cursor

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

arrow constant, busy constant, cross constant, defaultCursor property, hand constant, help constant, hotspot property, iBeam constant, lockCursor property, plus constant, watch constant, xHot property, yHot property, Why does a custom cursor fail to appear?, Development menu > Image Library

Summary

Specifies the shape of the cursor.

Syntax

set the cursor to {cursorName | imageID}

Example Code

```
set the cursor to watch
set the cursor to arrow
set the cursor to 21403
```

Comments

Use the cursor property to signal the status of a handler to the user or to indicate what kind of data the mouse pointer is over. For example, a watch cursor shows the user that a handler is executing, while an I-beam cursor indicates that the text under the mouse pointer is editable.

Value:

The cursor is the ID of an image to use for a cursor. Revolution looks for the specified image first in the current stack, then in other open stacks.

Revolution includes several built-in cursors whose names you can use in place of their image IDs.

Comments:

The built-in cursors and their recommended uses are:

- none: Hides the cursor
- busy: Use repeatedly during a long handler
- watch: Use during a moderately long handler

- arrow: Use for selecting objects
- cross: Use for painting, drawing, or selecting a point or small area
- hand: Use for clicking hypertext links
- iBeam: Use for selecting text in a field
- plus: Use for selecting items such as spreadsheet cells
- help: Use for getting online help

The busy cursor is a rotating beach ball. Each time you use the statement set the cursor to busy, the beach ball advances in its rotation. For example, the following statements cause the cursor to appear to spin as long as the repeat loop is running:

```
repeat until someCondition is true
  set the cursor to busy -- spins a bit further
  doSomething -- insert whatever you want the loop to do here
end repeat
```

You can also set the cursor property to the ID of an image. Custom cursor images must contain three colors: black, white, and a transparent color.

Cross-platform note: To be used as a cursor on Mac OS systems, an image must be 16x16 pixels. To be used as a cursor on Unix or Windows systems, an image must be 16x16 or 32x32 pixels.

If the lockCursor property is set to false, Revolution automatically sets the cursor according to its location once the handler finishes. (For example, the cursor normally turns into an arrow over a menu, an I-beam over an editable text field, and so on.) To retain the cursor property after a handler finishes, use the lock cursor command.

You can force Revolution to use the operating system's cursors with the following two statements:

```
delete stack "revCursors"
reset cursors
```

Caution! If you use the delete stack command to remove the "revCursors" stack, Revolution's cursors are permanently deleted and you will need to download a new cursors stack to restore them.

If you change the set of built-in Revolution cursors in the "revCursors" stack, you must either quit and restart the application or use the reset cursors command to begin using the new cursor shapes.

curve

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

choose command, graphic keyword, lineSize property, penColor property, penPattern property, style property, tool function, Object menu > New Control > Curve Graphic

Summary

Designates the paint tool used to draw curved lines. It also specifies, through the style property, that a graphic is a curved line.

Syntax

Example Code

```
choose curve tool  
set the style of last graphic to curve
```

Comments

Use the curve keyword to paint a curved line with the penColor or to change a graphic object to a curved line.

Comments:

When using the Curve tool, the cursor is a crosshairs shape (over images) or an arrow (anywhere else). This tool is used to draw a freehand line in the penColor (or penPattern).

If you try to paint with the Curve tool on a card that has no images, an image the size of the card is created to hold the painting. If the current card already has one or more images, painting outside the image has no effect.

Setting the style of a graphic to curve makes the graphic into a curved line. Curve graphics, unlike painted curves, can be changed and reshaped: use the points property to change the shape of the curve.

customKeys

property

Synonyms

userProperties

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

customProperties property, customPropertySet property, getProp control structure, keys function, properties property, set command, setProp control structure, About custom properties and custom property sets, How to create a custom property, How to delete a custom property, How to find out whether a custom property exists, How to list an object's custom properties, How to refer to a custom property in a non-active set, How to rename a custom property, Why doesn't a custom property appear in the property inspector?

Summary

Lists the names of all the custom properties of an object.

Syntax

set the customKeys of object to {propertiesList | empty}

Example Code

```
set the customKeys of stack "Preferences" to empty  
set the customKeys of last button to the customKeys of button 1
```

Comments

Use the customKeys property to find out what custom properties an object has, and to create or delete custom properties.

Value:

The customKeys of an object is a list of that object's custom property names, one per line.

Comments:

The customKeys lists the names of properties in the object's current customPropertySet. If the object has more than one custom property set, you must switch to the desired property set before checking the customKeys. The following example displays the custom properties in a custom property set called "Francois":

```
set the customPropertySet of button "My Button" to "Francois"  
answer the customKeys of button "My Button" -- in "Francois" set
```

If you set the customKeys of an object, and the propertiesList contains a custom property that the object doesn't have, a custom property with that name is created.

(A property name can be up to 255 characters long. The number of property names is limited only by Revolution's total memory space.) The value of a newly-created custom property is empty.

If you set the customKeys of an object and do not include the name of an existing custom property, that custom property is deleted. Setting the customKeys to empty deletes all custom properties in the current custom property set.

customProperties

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

customKeys property, customPropertySet property, customPropertySets property, getProp control structure, keys function, set command, setProp control structure, About containers, variables, and sources of value, About custom properties and custom property sets, How to create a custom property, How to delete a custom property, How to find out whether a custom property exists, How to list an object's custom properties, How to refer to a custom property in a non-active set, How to rename a custom property, How to store an array in a custom property

Summary

Specifies all the custom properties of an object that are in the current customPropertySet, along with their settings.

Syntax

set the customProperties of object to propertiesArray

set the customProperties[propertySet] of object to propertiesArray

Example Code

```
set the customProperties of this card to myPropertiesArray  
put the customProperties["mySet"] of me into myArray
```

Comments

Use the customProperties property to set or retrieve all the custom properties of an object at once.

Value:

The customProperties of an object is an array of custom properties and their values. The name of each custom property is the array key.

Comments:

The customProperties specifies the properties in the object's current customPropertySet. (The object can have other custom properties, which are accessed by switching to another customPropertySet.)

You can get or change the value of a single custom property using array notation. For example, the following statement copies the entire set of custom properties from one card to another, changing only the custom property named "changedDate":

```
put the customProperties of this card into myCustomProps
put field 3 into myCustomProps["changedDate"]
set the customProperties of next card to myCustomProps
```

Note: You can also set custom properties individually using the set command.

To refer to a custom property set that is not the current customPropertySet, use array notation to refer to the set. For example, to get an array consisting of all the custom properties in a custom property set called "MyProps", use a statement like the following:

```
get the customProperties["MyProps"] of this card
```


customPropertySet

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

customKeys property, customProperties property, customPropertySets property, getProp control structure, set command, setProp control structure, About custom properties and custom property sets, How to create a custom property set, How to delete a custom property set, How to duplicate a custom property set, How to refer to a custom property in a non-active set, How to rename a custom property set

Summary

Specifies a set of custom properties applied to an object.

Syntax

set the customPropertySet of object to {setName | empty}

Example Code

```
set the customPropertySet of me to "Mac Properties"
```

Comments

Use the customPropertySet property to create and switch between sets of custom properties and property values.

Value:

The customPropertySet of an object is the name of the active property set.

By default, the customPropertySet of an object is empty.

Comments:

An object can have multiple sets of custom properties. Each set is independent of the others, and the same property can have different values in different sets. To make a set's custom properties accessible, set the object's customPropertySet property to the name of the set you want to use. The values and properties in the current customPropertySet are used when you refer to a custom property.

If the setName is not already a custom property set, setting the customPropertySet to the setName creates a custom property set named setName for the object.

To use the default set of custom properties, set the `customPropertySet` to empty.

You can access a property that is not part of the current `customPropertySet` using array notation. For example, the following statement gets the value of a custom property named "foo" in a custom property set named "myCustomSet":

get the `myCustomSet["foo"]` of field "Example"

customPropertySets

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

customKeys property, customPropertySet property, getProp control structure, set command, setProp control structure, About custom properties and custom property sets, How to create a custom property set, How to delete a custom property set, How to duplicate a custom property set, How to refer to a custom property in a non-active set, How to rename a custom property set, Why doesn't a custom property appear in the property inspector?

Summary

Lists all the custom property sets that can apply to an object.

Syntax

get the customPropertySets of object

Example Code

```
if it is among the lines of the customPropertySets then doItOver
```

Comments

Use the customPropertySets property to find out what property sets are defined for an object.

Value:

The customPropertySets of an object is a list of all the custom property sets belonging to the specified object, one per line.

Comments:

You can delete an object's custom property sets by setting the object's customPropertySets to empty.

cut
command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clipboard function, clone command, copy command, delete command, paste command, selectedObject function, selection keyword, How to move a card between stacks, Edit menu > Cut

Summary

Copies the selected text or the selected section of an image, or an object, and deletes it from the selection.

Syntax

cut [object]

Example Code

```
cut -- cuts whatever is selected
cut player "Quivering Apple" from recent card
cut the selectedObject
```

Comments

Use the cut command to place objects or text on the clipboard while removing them from the stack.

Parameters:

The object is any available object. If no object is specified, the cut command cuts whatever is currently selected to the clipboard.

Comments:

If no object is specified, the cut command is equivalent to choosing Edit menu Cut.

cutKey

message

Synonyms

Objects

field, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backspaceKey message, copyKey message, cut command, delete command, deleteKey message, pasteKey message, undoKey message

Summary

Sent when the user presses the key combination equivalent to the Cut menu item.

Syntax

cutKey

Example Code

```
on cutKey
  if word 1 of the focusedObject is "card" then
    answer error "Cannot cut a card."
  else pass cutKey
end cutKey
```

Comments

Handle the cutKey message if you want to change the normal cut process, or prevent use of the Cut keyboard equivalent without changing the menu.

Comments:

The Revolution development environment traps the cutKey message. This means that the cutKey message is not received by a stack if it's running in the development environment. To test a cutKey handler, choose Development menu>Suspend Development Tools.

The cutKey message is sent when the user presses Command-X (on Mac OS systems), Control-X (on Windows systems), Shift-Delete (on Unix systems), or the keyboard Cut key.

The message is sent to the active (focused) control, or to the current card if no control is focused.

dashes

property

Synonyms

Objects

graphic, global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

angle property, filled property, foregroundColor property, foregroundPattern property, lineSize property, penColor property, penPattern property

Summary

Specifies the appearance of dashed lines in graphics and paint images.

Syntax

set the dashes [of graphic] to pixelsOn[,pixelsOff]

Example Code

```
set the dashes to 10,2 -- 10-pixel dashes separated by 2 pixels
set the dashes of graphic "Connector" to 1,2,5,2 -- "dit-dot" dashes
set the dashes of graphic 10 to empty -- makes a solid line
```

Comments

Use the dashes property to change the appearance of lines.

Value:

The dashes of a graphic is a list of an even number of positive integers.

By default, the dashes property of a newly created graphic is set to empty.

Comments:

The odd-numbered items in the dashes property represent the number of pixels in a dash, and the even-numbered items represent the number of pixels in the blank space after the dash. The list is repeated for the length of the line. For example, if the dashes is set to 20,10, the line consists of 20-pixel-long dashes separated by 10-pixel blanks.

If the dashes property contains a single integer or is empty, the line is solid.

The global setting of the dashes property controls the appearance of lines drawn with the paint tools. Once a paint line is drawn, its appearance cannot be changed by changing the global dashes property.

Cross-platform note: On Mac OS and OS X, and Windows 95/98 systems, the dashes property cannot control the length of dashes and blanks: setting the dashes to any non-empty value at all creates a dashed line, but the length of dashes and blanks is controlled by the operating system.

date

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

centuryCutoff property, convert command, dateFormat function, english keyword, internet keyword, milliseconds function, seconds function, system keyword, ticks function, time function, Recipe for storing a modification timestamp

Summary

Returns the current date.

Syntax

the [long | abbr[rev[iated]] | short] [english | system | internet] date
the [internet] [english | system] date
date()

Example Code

```
answer the date  
put the internet date into dateHeader  
put the long system date into field "Today's Date"
```

Comments

Use the date function to display the current date to the user, or to store the date for later use.

Value:

If the useSystemDate property is set to true or if you specify the system date, the times returned by the date function are formatted according to the user's system preferences. In general, the short date is briefer than the abbreviated date, which is briefer than the long date.

If the useSystemDate is false or if you specify the english time, the short, abbreviated, and long forms of the date are in the format described below:

The date form returns the month number, the day of the month, and the last two digits of the year, separated by slashes (/).

The short date form returns the same value as the date form.

The abbreviated date form returns the first three letters of the day of the week, a comma, the first three letters of the month, a space, the day of the month, a comma, and the year.

The long date form returns the day of the week, a comma, the name of the month, a space, the day of the month, a comma, and the year.

The internet date form returns the following parts, separated by spaces:

- the day of the week followed by a comma
- the day of the month
- the three-letter abbreviation for the month name
- the four-digit year
- the time in 24-hour format, including seconds, delimited by colons
- the four-digit time zone relative to UTC (Greenwich) time

Comments:

The format of the system date forms is set by the Date & Time control panel (on Mac OS systems), the Date control panel (on Windows systems), or the LANG environment variable (on Unix systems).

Changes to Transcript:

The ability to use the date format preferred by the user was introduced in version 1.1. In previous versions, the date function, along with the time function, consistently used the standard U.S. format, even if the operating system's settings specified another language or date format.

dateFormat

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

abbreviated keyword, convert command, date function, english keyword, long keyword, monthNames function, short keyword, system keyword, useSystemDate property, weekdayNames function

Summary

Returns a string, in the format of the C "strftime()" function, describing an available date format.

Syntax

the [long | abbr[rev[iated]] | short] dateFormat

dateFormat()

Example Code

```
put the long dateFormat into stringToParse
```

Comments

Use the dateFormat function to obtain the general format of the date for further processing.

Value:

The dateFormat function returns a string.

Comments:

The dateFormat function returns a string containing one or more formatting incantations, each of which describes a part of the requested date format. The possible incantations are as follows:

%a

Abbreviated weekday name: the abbreviated day of the week, as reported by the weekdayNames function

%A

Long weekday name: the long day of the week, as reported by the weekdayNames function

%b

Abbreviated month name: the abbreviated month name, as reported by the monthNames function

`%B`

Long month name: the long month name, as reported by the `monthNames` function

`%d`

Day of the month: the day of the month as a number

`%m`

Month number: the number of the month

`%y`

Two-digit year: the year as a two-digit number

`%Y`

Four-digit year: This incantation indicates the year as a four-digit number (including the century)

The `dateFormat` function always returns the date format according to the user's current system preferences, regardless of the setting of the `useSystemDate` property.

dateItems

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

convert command, date function, seconds keyword, time function

Summary

Designates a comma-delimited date format used with the convert command.

Syntax

Example Code

```
convert myVariable to dateItems  
convert it from dateItems to long date and long time
```

Comments

Use the dateItems keyword with the convert command to store a date and/or time.

Comments:

The dateItems format is a comma-separated list of numbers:

- ĩ the year
- ĩ the month number
- ĩ the day of the month
- ĩ the hour in 24-hour time
- ĩ the minute
- ĩ the second
- ĩ the numeric day of the week where Sunday is day 1, Monday is day 2, and so forth

The dateItems does not change depending on the user's settings, so you can use it (or the seconds format) to store a date and time with a stack, in an invariant form that won't change.

dateTime

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

convert command, date function, numeric keyword, sort command, sort container command, text keyword, time function

Summary

Used with the sort and sort container commands to sort by date and/or time.

Syntax

Example Code

```
sort cards dateTime by field "Last Modified Date"
```

Comments

Use the dateTime keyword when sorting by a field or portion of a container that is a date or time.

Comments:

Alphabetical sorting does not take the special format of dates and times into account. For example, a normal sort places 11:30 AM after 2:00 AM, because 2 is greater than 1. The dateTime keyword recognizes all Transcript's date and time formats and sorts them in time order, rather than alphabetical or numeric order.

debugDo

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

do command, send command, traceAbort property, traceDelay property, traceReturn property, traceStack property

Summary

Executes a list of statements with additional debugging information when in the debugger.

Syntax

debugDo statementList

Example Code

```
debugDo "beep"  
debugDo commandsList
```

Comments

The debugDo command is used by Revolution's message box and debugger.

Parameters:

The statementList is a Transcript statement, a container with one or more statements, or a string that evaluates to a statement.

Comments:

Normally, you should use the do command; consider using debugDo only if you are writing a debugger or similar utility, or if you need to debug the statement list sent by a do command.

When the debugger is active, you can use debugDo to access local variables using the message box. If you enter a debugDo statement into the message box, and the statement refers to a local variable, the variable's value is the current value in the handler that the debugger is executing. This capability is useful if you want to track the current value of a variable in a handler you're debugging.

Important! The details of this command may change in future releases.

decompress

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

base64Decode function, compress function, URLDecode function

Summary

Returns the plain text of a gzip-encoded string.

Syntax

decompress(gzippedString)
the decompress of gzippedString

Example Code

```
decompress(receivedString)
put decompress(it) into URL "file:data.txt"
go stack decompress(URL "binfile:newstuff.gz")
```

Comments

Use the decompress function to regain the original data that was compressed.

Parameters:

The data is a string of compressed binary data.

Value:

The decompress function returns a string.

Comments:

The decompress function is the inverse of the compress function.

The uncompressed result is typically about half again the size of the compressed data, although different results may be obtained depending on the amount of data.

For technical information about the format used by the compress and decompress functions, see RFC 1952 at <<http://www.ietf.org/rfc/rfc1952.txt>>. The decompress function uses the zlib compression library.

decorations

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

closeBox property, closeStackRequest message, default keyword, maximize keyword, menu keyword, metal property, minimize keyword, modal command, mode property, modeless command, palette command, resizable property, shadow property, style property, systemWindow property, title keyword, topLevel command, windowID property, windowShape property, zoomBox property, How to prevent the user from moving a window, How to prevent the user from resizing a window, How to remove a window's close box, How to remove a window's title bar

Summary

Specifies the window controls and appearance of a stack window.

Syntax

set the decorations of stack to {controlList | WDEF | default | empty}

Example Code

```
set the decorations of this stack to "title,minimize"
set the decorations of stack "Splash Screen" to empty
set the decorations of stack myStack to 127 -- a WDEF resource
```

Comments

Use the decorations property to change the appearance of windows.

Value:

The decorations of a stack is one of the following:

- "default"
- empty
- one or more of "title", "minimize", "maximize", "close", "menu", "system", "noShadow", and "metal", separated by commas
- a WDEF identifier (on Mac OS or OS X systems)

By default, the decorations property of newly created stacks is set to "default".

Comments:

You can specify one or more of the following decorations:

- `title`: Shows the window's title bar
- `minimize`: Shows the minimize (iconify) box in the title bar
- `maximize`: Shows the maximize (zoom) box in the title bar
- `close`: Shows the close box in the title bar
- `menu`: Shows the menu bar in the window (Unix and Windows only)
- `system`: Shows the window as a system window (OS X, Unix, and Windows only)
- `noShadow`: Remove the window's drop shadow (OS X only)
- `metal`: Shows the window with a textured appearance (OS X only)

Note: The "metal" and "system" options are not compatible. If you set a stack's decorations to a string that includes both, its window becomes a non-metal system window.

Cross-platform note: On Mac OS systems, the "system" option has no effect.

Setting a stack's decorations to empty removes the entire title bar, along with the window borders.

Setting a stack's decorations to "default" gives it appropriate decorations for its style. If the style is "topLevel", the window includes all the decorations (title bar, menu bar [on Unix and Windows systems], close box, minimize box, maximize box) by default. Otherwise, the window includes a title bar by default.

The decorations property interacts with the `closeBox`, `minimizeBox`, `metal`, `shadow`, `systemWindow`, and `zoomBox` properties. Setting any of these properties to true changes the decorations property to include the corresponding option, and including any of the "close", "minimize", "metal", "noShadow", "system", and "maximize" options in a stack's decorations sets the corresponding property to true.

Important! The decorations are set all at once, and if you don't include a decoration, it is removed from the stack window. For example, if you set the decorations to "metal", the resulting window won't include a title bar, even if it had one initially. To set a stack to metal appearance while showing a title bar with close, minimize, and maximize box, you need to set all the decorations at once:

set the decorations of stack "Test" to

```
"metal,title,close,minimize,maximize"
```

This applies only to the decorations property. If you use the individual properties (such as `metal` and `closeBox`) instead, you don't need to reset the other decorations-related properties.

Cross-platform note: On Mac OS and OS X systems, the "menu" decoration has no effect. On Windows systems, the "menu" decoration must be set along with the "maximize" or "minimize" decorations: you cannot use "maximize" or "minimize" without including "menu". On OS X systems, the minimize, close, and zoom boxes are always visible if the title bar is, and turning off these options disables the boxes instead of removing them, in conformance with user-interface guidelines for this platform.

A WDEF identifier is the number of a Mac OS WDEF resource multiplied by 16, plus a variation code between zero and 15. For example, if you have a WDEF resource whose ID is 124, and you want to use

its 14th variation (which is number 13, since the numbers start with zero), set the stack's decorations property to 1997. This number is 124 (the resource ID) times 16, plus 13 (the variation code). Various WDEF resources are included in the operating system—the exact WDEFs available vary from version to version. WDEF routines, like externals, can be written in a programming language and compiled.

Cross-platform note: WDEF resources can be used only on Mac OS and OS X systems. For a cross-platform method of changing a window's shape, use the `windowShape` property.

Changes to Transcript:

The "close", "system", "noShadow", and "metal" options were introduced in version 2.1.

default

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

case keyword, decorations property, default property, maximize keyword, minimize keyword, switch control structure, titlebar keyword, Recipe for a Cards menu

Summary

Used with the decorations property to specify the standard appearance for a window type. When used within a switch control structure, specifies a section of statements to be executed in all cases.

Syntax

Example Code

```
set the decorations of stack "Main" to default
```

Comments

Use the default keyword to set a stack window to a normal window appearance.

Comments:

An editable window whose decorations property is set to default has a title bar, close box, zoom box, and collapse box (on Mac OS systems), or a title bar, minimize button, maximize button, and menu bar (on Unix and Windows systems). A palette, modal, or modeless window whose decorations is set to default has a title bar.

If a switch control structure has a default section, the statements in the default section are executed if no break statement has been encountered yet in the switch control structure. (Usually, there is a break statement within each case section, so the default section is executed only if none of the case conditions is true.)

default property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

acceleratorKey property, defaultButton property, dialogData property, enterInField message, enterKey message, returnInField message, returnKey message, style property, How to make a throbbing button, How to make the Return or Enter key activate a button

Summary

Specifies that a button will be automatically chosen when the user presses Return or Enter.

Syntax

set the default of button to {true | false}

Example Code

```
set the default of button "OK" to true  
set the default of button otherButton to false
```

Comments

Use the default property when designing cards to be used as dialog boxes. The familiar appearance of the default button is a cue to users about what to expect when they use the shortcut of pressing Return or Enter.

Value:

The default of a button is true or false.

By default, the default property of newly created buttons is set to false.

Comments:

When the user presses Enter or Return and there is no active control, Revolution sends a mouseUp message to the button whose default property is true. (Also handle the returnInField and enterInField messages to ensure that the default button is activated even if there is a text insertion point or a control is focused.)

If more than one button's default is true, the message is sent to the button whose default property was set to true most recently.

Changing a button's default property increases its size, so it also changes its rectangle (and related properties). Changing the default property does not change the button's location.

Cross-platform note: On OS X systems, the default button throbs rhythmically instead of having a border. Because the throbbing button is drawn by the operating system, its appearance may not be correct if the button is placed on top of another control. To avoid minor cosmetic problems, place default buttons directly on the card rather than on top of other objects.

defaultButton

property

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

default property, enterInField message, enterKey message, ID property, returnInField message, returnKey message

Summary

Reports the long ID of the current card's default button.

Syntax

get the defaultButton of card

Example Code

```
send mouseUp to the defaultButton of this card
```

Comments

Use the defaultButton property to find out which button on a card is designated as the button to be clicked when the user presses the Return or Enter key.

Value:

The defaultButton of a card is the long ID property of a button.

This property is read-only and cannot be set.

Comments:

The defaultButton property returns the long ID of the button whose default property is set to true.

When the user presses Enter or Return and there is no active control, Revolution sends a mouseUp message to the defaultButton. (If the card contains an editable field, handle the returnInField and enterInField messages to ensure that the default button is activated.)

If more than one button's default is true, the defaultButton property reports the ID of the button whose default property was set to true most recently. This property reports empty if there is no default button.

defaultCursor

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cursor property, lockCursor property, Why does a custom cursor fail to appear?, Development menu > Image Library

Summary

Specifies the default shape of the cursor.

Syntax

set the defaultCursor to imageID

Example Code

```
set the defaultCursor to arrow
```

Comments

Use the defaultCursor property to change the cursor used when the Browse tool is being used from a hand to some other cursor shape.

Value:

The defaultCursor is the ID of an image to use for a cursor. Revolution looks for the specified image first in the current stack, then in other open stacks.

Tip: Revolution includes several built-in cursors whose names you can use in place of their image IDs. The built-in cursors are described in the entry for the cursor property.

Comments:

If the lockCursor property is set to false, the cursor changes shape automatically as the mouse moves. (For example, the cursor normally turns into an arrow over a menu, an I-beam over an editable field, and so on.) The defaultCursor is the shape used automatically when the mouse pointer is in a stack window, but not in an editable field.

defaultFolder

property

Synonyms

folder, directory

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer folder command, create folder command, delete file command, revSetDatabaseDriverPath command, specialFolderPath function, About filename specifications and file paths, How to change the current folder used for file operations, How to determine whether a file exists, How to find the Preferences folder, How to find out the location of the current stack's file, How to get the location of the user's home directory, Recipe for converting an absolute path to a relative path, Recipe for converting a relative path to an absolute path

Summary

Specifies the folder that is used by the files and folders functions and by commands that manipulate files.

Syntax

set the defaultFolder to folderPath

Example Code

```
set the defaultFolder to "/Hard Disk/Applications/GetIt"  
set the defaultFolder to it
```

Comments

Use the defaultFolder to perform file manipulations on files in the same folder without having to include the full path.

Value:

The defaultFolder is a string consisting of a valid path to a folder.

When you start up the application, the defaultFolder is set to the folder containing the application. If you're using the development environment, this is the folder containing Revolution; if you're using a standalone application, this is the folder containing that standalone. (On OS X systems, the defaultFolder is set to the folder that contains the application bundle.)

Comments:

The defaultFolder property specifies the folder Revolution uses as the current directory when resolving relative paths (except for relative paths specified in the stackFiles property).

If you specify a file without giving its full path, Revolution looks for the file in the defaultFolder. If you specify a relative path, the defaultFolder is prepended to it to create the full path.

If you set the defaultFolder to a folder that doesn't exist, the result is set to "can't open directory" and the value of the defaultFolder does not change.

You cannot delete the current defaultFolder.

Important! The folderPath is specified using Unix conventions, even on Mac OS and Windows systems. The names of folders are separated with a "/" character, and absolute paths (starting with a disk or partition name) must begin with a "/" character.

defaultMenubar

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

defaultStack property, editMenus property, menubar property, About menus and the menu bar, How to switch between menu bars, Tools menu > Menu Builder

Summary

Specifies which menu bar is displayed on Mac OS systems when the active window's menubar property is empty.

Syntax

set the defaultMenubar to groupName

Example Code

```
set the defaultMenubar to "Standard Menus"
```

```
set the defaultMenubar to the short name of group 1 of stack "Menubars"
```

Comments

Use the defaultMenubar property to change the menus used by stacks that don't have their own custom menu bar.

Value:

The defaultMenubar is the short name of any group that is in a stack that is in memory.

Comments:

The defaultMenuBar specifies a group of buttons. Each button in the group becomes a menu in the menu bar.

The setting of the defaultMenubar property has no effect on Unix and Windows systems, except that references to menus are taken as referring to the defaultMenubar if the stack in the active window has no menu bar of its own.

defaultStack

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

defaultMenuBar property, focusedObject function, send command, topStack function, About object types and object references

Summary

Specifies which stack the expression this stack evaluates to, if no stack is specified in a statement.

Syntax

set the defaultStack to stackName

Example Code

```
set the defaultStack to "Help"  
set the defaultStack to the topStack
```

Comments

Use the defaultStack property to ensure that actions are performed on the correct stack.

Value:

The defaultStack is the name of an open stack.

Comments:

If no stack is specified in a statement that applies to a stack, Revolution assumes the defaultStack is the stack meant. For example, the following statement causes the defaultStack to move to its next card:

```
go next card -- of the current defaultStack
```

The expression this card refers to the current card of the defaultStack.

Important! You cannot use a stack reference, only a stack name or an expression that evaluates to a stack name. For example, the following statement causes an error:

```
set the defaultStack to stack "Hmm" -- WON'T WORK
```

but the following statements are correct:

```
set the defaultStack to "Hmm"
```

```
set the defaultStack to ("stack" && quote & "Hmm" & quote)
```

The defaultStack property is particularly useful in stacks opened in a mode other than an editable window (such as stacks that are being used as dialog boxes, palettes, or menus). Revolution's message box and editing palettes set the defaultStack property to the value returned by the topStack function before performing a stack action.

define

command

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

customKeys property, set command, undefine command

Summary

Has no effect and is included in Transcript for compatibility with imported SuperCard projects.

Syntax

define property of object

Example Code

Comments

In SuperCard and Oracle Media Objects, the define command is used to create a user property.

In Revolution, you can use a custom property simply by setting its value with the set command, without creating it first, so this command is not needed.

delete

command

Synonyms
clear

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

cut command, delete chunk command, delete file command, delete stack command, delete variable command, deleteBackground message, deleteButton message, deleteCard message, deleteEPS message, deleteField message, deleteGraphic message, deleteGroup message, deleteImage message, deletePlayer message, deleteScrollbar message, remove command, undo command, Why does a repeat loop behave strangely?, Recipe for getting the dimensions of a picture file, Edit menu > Clear

Summary

Removes the selected text, or the selected section of an image, or an object.

Syntax

delete [object]

Example Code

```
delete  
delete this card  
delete button "New Button"
```

Comments

Use the delete command to remove objects or text from the stack.

Parameters:

The object is any available object. If no object is specified, the delete command removes whatever is currently selected.

Comments:

If no object is specified, the delete command is equivalent to pressing the Delete key. It removes the selection.

Use the delete command to remove a control, card, or group permanently. To remove an object and place it on the clipboard for pasting elsewhere, use the cut command instead. To temporarily hide a control, use the hide command instead.

Important! You cannot delete the object whose script holds a currently executing handler.

delete chunk

command

Synonyms

clear chunk

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete command, delete file command, delete stack command, delete variable command, filter command, put command, About chunk expressions, How to delete a line from a container, Recipe for a scrolling text banner, Recipe for converting an absolute path to a relative path, Recipe for converting a relative path to an absolute path

Summary

Removes text from a container.

Syntax

delete [chunk of] container

Example Code

```
delete word 2 of field "Instructions"  
delete line 6 to 8 of myCompiledVariable  
delete message -- removes content of the message box, not the box
```

Comments

Use the delete chunk command to change the contents of a container.

Parameters:

The chunk is a chunk expression specifying a portion of the container.

The container is a field, button, or variable, or the message box.

Comments:

The delete chunk command is equivalent to the following statement:

```
put empty into chunk of container
```


delete file

command

Synonyms
clear file

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close file command, delete chunk command, delete command, delete folder command, delete stack command, delete URL command, delete variable command, deleteResource function, open file command, rename command, sysError function, How to delete a file

Summary

Removes a file from the user's system.

Syntax

delete file pathname

Example Code

```
delete file "/tmp/handlers.txt"  
delete file "My Test.rev"
```

Comments

Use the delete file command to clean up by removing a file you created.

Parameters:

The pathName specifies the name and location of the file. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

Comments:

This command can also be used to remove files your stack did not create. Of course, a stack should not remove files and folders it didn't create without obtaining explicit confirmation from the user.

If the file doesn't exist, is a folder, or for some other reason cannot be deleted, the result is set to "Can't delete that file."

Caution! This command cannot be undone, so be very certain before you use it. The delete file command removes the file completely from the user's system. It does not place the file in the Trash or Recycle Bin.

delete folder

command

Synonyms

clear folder, delete directory, clear directory

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete command, delete file command, delete stack command, delete URL command, rename command, sysError function

Summary

Removes a folder from the user's system.

Syntax

delete folder pathname

Example Code

```
delete folder "/bin/badprogram/"
delete folder "../Last Version"
delete folder (the foundText)
```

Comments

Use the delete folder command to clean up by removing a folder you created.

Parameters:

The pathName specifies the name and location of the folder. If you specify a name but not a location, Revolution assumes the folder is in the defaultFolder.

Comments:

If a folder contains any files or folders, it cannot be deleted with the delete folder command until the items inside it are deleted or moved. To delete a folder that is not empty, use the revDeleteFolder command.

This command can also be used to remove folders your stack did not create. Of course, a stack should not remove folders it didn't create without obtaining explicit confirmation from the user.

Caution! This command cannot be undone, so be very certain before you use it. The delete folder command removes the folder completely from the user's system. It does not place the folder in the Trash or Recycle Bin.

delete stack

command

Synonyms

clear stack

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantDelete property, delete chunk command, delete command, delete file command, delete variable command, deleteStack message, destroyStack property, File menu > Close and Remove From Memory...

Summary

Removes a substack from a stack file, or removes a main stack from memory.

Syntax

delete stack

Example Code

```
delete stack "Expressions"
```

Comments

Use the delete stack command to delete a substack, or to purge a main stack from memory.

Parameters:

The stack is any valid stack reference.

Comments:

The delete stack command has different effects, depending on whether the stack is a main stack or a substack.

If the stack is a main stack, the delete stack command removes the stack from memory. (Any unsaved changes are lost.) However, it does not remove it from the user's system. To delete a main stack, use the delete file command instead.

Caution! If the stack is a substack, the delete stack command deletes the stack from memory, which causes it to be deleted permanently the next time its stack file is saved.

delete URL

command

Synonyms
clear URL

Objects
Internet library

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
binfile keyword, delete file command, delete folder command, file keyword, ftp keyword, http keyword, libURLSetLogField command, resfile keyword, URL keyword, How to access the Internet from behind a firewall, How to remove a file from an FTP server

Summary
Removes a file or directory from a server, or removes a file from your system.

Syntax
delete URL URLtoDelete

Example Code

```
delete URL "ftp://root:secret-word@mars.example.org/deleteable.txt"  
delete URL "ftp://me:mine@ftp.example.net/trash/"  
if the hilite of button "Remove" then delete URL (field "Location")
```

Comments
Use the delete URL command to remove a file.

Parameters:
The URLtoDelete specifies the name and location of the file or directory to delete, in the form of a URL.

Comments:
If the file or directory is successfully deleted, the result function is set to empty. Otherwise, the result function returns an error message.

You can use a file or binfile URL to delete a file, but not a folder. To remove a folder from your system, use the delete folder command. (You can also use the delete file command to delete a local file.)

This command can also be used to remove files and directories your stack did not create. Of course, a stack should not remove data it didn't create without obtaining explicit confirmation from the user.

Normally, FTP servers do not allow anonymous users to delete files, for obvious reasons. This means that while an ftp URL without a user name and password is valid, you will almost always need a user name and password to use the delete URL command.

Note: When used with an ftp or http URL, the delete URL command is blocking: that is, the handler pauses until Revolution is finished deleting the URL. Since deleting a file from a server may take some time due to network lag, the delete URL command may take long enough to be noticeable to the user.

Important! If a blocking operation involving a URL (using the put command to upload a URL, the post command, the delete URL command, or a statement that gets an ftp or http URL) is going on, no other blocking URL operation can start until the previous one is finished. If you attempt to use a URL in an expression, or put data into a URL, while another blocking URL operation is in progress, the result is set to "Error Previous request not completed".

Important! The delete URL command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the delete URL command is not yet available. This may affect attempts to use this command in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the delete URL command can be used in any handler.

delete variable

command

Synonyms

clear variable

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete chunk command, delete command, delete file command, delete stack command, global command, globalNames function, intersect command, local command, variableNames function, About containers, variables, and sources of value, How to delete a custom property

Summary

Removes a variable from memory.

Syntax

delete {local | global | variable} {variableName | arrayIndex}

Example Code

```
delete local tempVariable  
delete global myArray[17] -- removes 17th element of that array
```

Comments

Use the delete variable command to free memory used by a large variable, or to clean up after using many variable names.

Parameters:

The variableName is the name of any local or global variable.

The arrayIndex is a key of an array variable. If an arrayIndex is specified instead of a variable name, the delete variable command removes that element of the array, without deleting the rest of the elements in the array.

Comments:

If you use the delete variable form, the variableName, global or local, is deleted.

The delete variable command not only removes the contents of the variable, but deletes it entirely from memory. If you delete a key from an array variable, that element of the array no longer exists as part of the variable.

Note: Local variables that are used within a handler are automatically deleted when the handler in which they are used exits.

deleteBackground

message

Synonyms

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantDelete property, delete command, deleteCard message, deleteGroup message, newBackground message

Summary

Sent to a group just before it is removed from the stack.

Syntax

deleteBackground

Example Code

```
on deleteBackground
  beep 2 -- warn user
  pass deleteBackground
end deleteBackground
```

Comments

Handle the deleteBackground message if you want to perform cleanup before a group is removed from the stack.

Comments:

The actual deletion is not triggered by the deleteBackground message, so trapping the message and not allowing it to pass does not prevent the group from being removed.

Deleting the last card on which a group appears does not remove the group from the stack, so it does not cause a deleteBackground message to be sent.

Ungrouping a group does not cause a deleteBackground message to be sent.

A deleteGroup message is sent before the deleteBackground message. (The deleteBackground message is included for compatibility with HyperCard.)

deleteButton

message

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cut command, delete command, newButton message

Summary

Sent to a button just before it is removed from the stack.

Syntax

deleteButton

Example Code

```
on deleteButton -- prevent the deletion by immediately undoing it
  beep
  send "undo" to this card in 5 milliseconds
end deleteButton
```

Comments

Handle the deleteButton message if you want to perform cleanup before a button is removed from the stack.

Comments:

The actual deletion is not triggered by the deleteButton message, so trapping the message and not allowing it to pass does not prevent the button from being removed.

However, the undo command will restore a button after it is deleted by the user.

deleteCard

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantDelete property, delete command, deleteStack message, newCard message

Summary

Sent to a card just before it is removed from the stack.

Syntax

deleteCard

Example Code

```
on deleteCard -- update a list of cards to reflect the deletion
  send "updateList" to this stack in 10 ticks -- after card is deleted
  pass deleteCard
end deleteCard
```

Comments

Handle the deleteCard message if you want to perform cleanup before a card is removed from the stack.

Comments:

The actual deletion is not triggered by the deleteCard message, so trapping the message and not allowing it to pass does not prevent the card from being removed.

deleteEPS

message

Synonyms

Objects

EPS

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete command, newEPS message

Summary

Sent to an EPS object just before it is removed from the stack.

Syntax

deleteEPS

Example Code

```
on deleteEPS -- prevent the deletion by immediately undoing it
  beep
  send "undo" to this card in 5 milliseconds
end deleteEPS
```

Comments

Handle the deleteEPS message if you want to perform cleanup before an EPS object is removed from the stack.

Comments:

The actual deletion is not triggered by the deleteEPS message, so trapping the message and not allowing it to pass does not prevent the EPS object from being removed.

However, the undo command will restore an EPS object after it is deleted by the user.

deleteField

message

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete command, newField message

Summary

Sent to a field just before it is removed from the stack.

Syntax

deleteField

Example Code

```
on deleteField -- first save the contents in a custom property
  set the (the short name of the target) of this card to target
  pass deleteField
end deleteField
```

Comments

Handle the deleteField message if you want to perform cleanup before a field is removed from the stack.

Comments:

The actual deletion is not triggered by the deleteField message, so trapping the message and not allowing it to pass does not prevent the field from being removed.

However, the undo command will restore a field after it is deleted by the user. For example, the following handler, placed in a card or stack script, effectively prevents a field from being deleted by the user:

```
on deleteField
  beep
  send "undo" to this card in 5 milliseconds
end deleteField
```

deleteGraphic

message

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete command, newGraphic message

Summary

Sent to a graphic just before it is removed from the stack.

Syntax

deleteGraphic

Example Code

```
on deleteGraphic -- announce deletion to user
  answer "You just removed" && the name of the target
  pass deleteGraphic
end deleteGraphic
```

Comments

Handle the deleteGraphic message if you want to perform cleanup before a graphic is removed from the stack.

Comments:

The actual deletion is not triggered by the deleteGraphic message, so trapping the message and not allowing it to pass does not prevent the graphic from being removed.

However, the undo command will restore a graphic after it is deleted by the user. For example, the following handler, placed in a card or stack script, effectively prevents a graphic from being deleted by the user:

```
on deleteGraphic
  beep
  send "undo" to this card in 5 milliseconds
end deleteGraphic
```

deleteGroup

message

Synonyms

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantDelete property, delete command, deleteBackground message, deleteCard message, deleteStack message

Summary

Sent to a group just before it is removed from the stack.

Syntax

deleteGroup

Example Code

```
on deleteGroup -- remove the group being deleted from a list
  get lineOffset(the short name of the target,field "Groups List")
  delete line it of field "Groups List"
  pass deleteGroup
end deleteGroup
```

Comments

Handle the deleteGroup message if you want to perform cleanup before a group is removed from the stack.

Comments:

The actual deletion is not triggered by the deleteGroup message, so trapping the message and not allowing it to pass does not prevent the group from being removed.

Deleting the last card on which a group appears does not remove the group from the stack, so it does not cause a deleteGroup message to be sent.

Ungrouping a group does not cause a deleteGroup message to be sent.

A deleteBackground message is sent after the deleteGroup message.

deleteImage

message

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete command, newImage message

Summary

Sent to an image just before it is removed from the stack.

Syntax

deleteImage

Example Code

```
on deleteImage
  export the target to file "Saved"
  pass deleteImage
end deleteImage
```

Comments

Handle the deleteImage message if you want to perform cleanup before an image is removed from the stack.

Comments:

The actual deletion is not triggered by the deleteImage message, so trapping the message and not allowing it to pass does not prevent the image from being removed.

However, the undo command will restore an image after it is deleted by the user. For example, the following handler, placed in a card or stack script, effectively prevents an image from being deleted by the user:

```
on deleteImage
  beep
  send "undo" to this card in 5 milliseconds
end deleteImage
```

deleteKey

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backspaceKey message, cutKey message, delete command

Summary

Sent to the active (focused) control, or to the current card if there is no active control.

Syntax

deleteKey

Example Code

```
on deleteKey -- clear the entire field
  if word 1 of the name of the target is "field" then
    put empty into the target
  end if
end deleteKey
```

Comments

Handle the deleteKey message if you want to do something special when the user presses the forward delete key.

Comments:

The message is sent to the active (focused) control, or to the current card if no control is focused.

The forward delete key is not the same as the backspace or delete key:

- The forward delete key is usually to the right of the main keyboard layout, and may be labeled with the word "Delete" or a left-facing arrow. Pressing it sends a deleteKey message.

- The backspace key is usually on the main keyboard above the Return key, and is may be labeled "Backspace" or "Delete". Pressing it sends a backspaceKey message.

deletePlayer

message

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete command, newPlayer message

Summary

Sent to a player just before it is removed from the stack.

Syntax

deletePlayer

Example Code

```
on deletePlayer -- delete the corresponding movie file
  delete file (the filename of the target)
  pass deletePlayer
end deletePlayer
```

Comments

Handle the deletePlayer message if you want to perform cleanup before a player is removed from the stack.

Comments:

The actual deletion is not triggered by the deletePlayer message, so trapping the message and not allowing it to pass does not prevent the player from being removed.

However, the undo command will restore a player after it is deleted by the user. For example, the following handler, placed in a card or stack script, effectively prevents a player from being deleted by the user:

```
on deletePlayer
  beep
  send "undo" to this card in 5 milliseconds
end deletePlayer
```

deleteRegistry

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

platform function, queryRegistry function, setRegistry function

Summary

Removes an entry from the Windows system registry.

Syntax

deleteRegistry(keyPath)

Example Code

```
deleteRegistry("HKEY_CLASSES_ROOTtxt")
if not deleteRegistry(storedKey) then answer "Couldn't delete key."
```

Comments

Use the deleteRegistry function to uninstall a registry entry your application has previously installed.

Parameters:

The keyPath parameter is the path to a registry entry.

Value:

The deleteRegistry function returns true if the entry was successfully deleted, false otherwise.

Comments:

If Windows sends an error message to the application, the error message is returned in the result.

Tip: To delete a subkey, use the setRegistry function to set the subkey's value to empty.

On Mac OS and Unix systems, the deleteRegistry function returns "not supported".

Caution! Be careful to use only carefully debugged entries with the deleteRegistry function., and be very certain you know all the ramifications of the entry you're removing. Removing entries from the Windows registry can cause the system to behave unexpectedly or stop functioning altogether.

deleteResource

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

copyResource function, delete command, delete file command, externals property, getResources function, sysError function

Summary

Removes a resource from the resource fork of a Mac OS file.

Syntax

deleteResource(file,resourceType,{resourceID | resourceName})

Example Code

```
get deleteResource("New Build",ICN#,129)
put deleteResource(line x of theFiles,XFCN,"Colorize") into temp
```

Comments

Use the deleteResource function to delete a resource from a file.

Parameters:

The file is the name and location of the file containing the resource. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

The resourceType is the 4-character type of the resource you want to delete.

The resourceID is an integer that specifies the resource ID of the resource you want to delete.

The resourceName is the name of the resource you want to delete.

Value:

The deleteResource function always returns empty.

Comments:

The deleteResource function deletes individual resources within the file, but does not delete the resource fork.

If the file does not exist, the result is set to "can't find file".

If the file exists but has no resource fork, the result is set to "can't open resource fork".

If the file exists but does not contain the specified resource, the result is set to "can't find the resource specified".

If the file is already open, the result is set to "Can't open resource fork".

deleteScrollbar

message

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete command, newScrollbar message

Summary

Sent to a scrollbar just before it is removed from the stack.

Syntax

deleteScrollbar

Example Code

```
on deleteScrollbar -- prevent the deletion by immediately undoing it
    beep
    send "undo" to this card in 5 milliseconds
end deleteScrollbar
```

Comments

Handle the deleteScrollbar message if you want to perform cleanup before a scrollbar is removed from the stack.

Comments:

The actual deletion is not triggered by the deleteScrollbar message, so trapping the message and not allowing it to pass does not prevent the scrollbar from being removed.

However, the undo command will restore a scrollbar after it is deleted by the user.

deleteStack

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantDelete property, delete stack command, newStack message

Summary

Sent to the current card of a stack that's about to be deleted.

Syntax

deleteStack

Example Code

```
on deleteStack -- warn the user
  if the mainStack of the owner of the target

    is not the owner of the target then -- it's a substack
      answer "This stack will be permanently deleted the" &&

        "next time you save the file."
    end if
    pass deleteStack
  end deleteStack
```

Comments

Handle the deleteStack message if you want to perform cleanup before a stack is deleted from its stack file.

Comments:

The actual deletion is not triggered by the deleteStack message, so trapping the message and not allowing it to pass does not prevent the stack from being removed.

If a stack's file is deleted with the delete file command (or by the user), and the stack itself is not open, no deleteStack message is sent.

descending

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

ascending keyword, sort command

Summary

Used with the sort and sort container commands to sort in reverse order (from greater to less).

Syntax

Example Code

```
sort marked cards descending by the name of this card  
sort field 3 descending dateTime by item 2 of each
```

Comments

Use the descending keyword to sort backwards.

Comments:

The descending keyword reverses the normal sort order:

• A descending text sort puts Z first, A last.

• A descending numeric sort puts large numbers first, small numbers last.

• A descending dateTime sort puts later times first, earlier times last.

destroyStack

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close command, closeStack message, delete stack command, deleteStack message, destroyWindow property, reloadStack message, stacks function, Why am I running out of memory?, File menu > Close and Remove From Memory...

Summary

Specifies whether a stack is purged from memory when it's closed, or whether it remains in memory.

Syntax

set the destroyStack of stack to {true | false}

Example Code

```
set the destroyStack of this stack to true
```

Comments

Use the destroyStack property to leave a stack in memory after it is closed.

Value:

The destroyStack of a stack is true or false.

By default, the destroyStack property of newly created stacks is set to false.

Comments:

If a stack's destroyStack property is set to true, closing all stacks in its stack file removes it from memory space, cleaning up after the stack and freeing memory for use by the application. If any handlers in the stack are running, the stack is not purged from memory until all handlers have completed. (The stack is not removed from memory until all stacks in the same stack file are also closed.)

If the destroyStack is false, closing a stack's window leaves it in memory. If you open and close many stacks in a session, and the destroyStack of all these stacks is false, the memory used by these stacks will continue growing until you quit the application. If you reopen the stack during the same session, a warning message appears cautioning you that the stack is already open.

If you close a stack whose `destroyStack` is true while a handler is running, the stack is removed from memory after all running handlers finish executing. This means that if you close and re-open a stack during a handler, and the stack's `destroyStack` is true, the stack is removed from memory (and closed again) after the handler finishes.

Note: Despite its alarming name, the `destroyStack` property does not destroy or damage a stack. It simply allows it to be automatically removed from memory.

destroyWindow

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close command, closeStack message, delete stack command, deleteStack message, destroyStack property, reloadStack message, windowID property

Summary

Specifies whether the memory used by a stack window is purged when the stack is closed.

Syntax

set the destroyWindow of stack to {true | false}

Example Code

```
set the destroyWindow of stack "Test Harness" to true
```

Comments

Use the destroyWindow property when installing new externals, or to save memory.

Value:

The destroyWindow of a stack is true or false.

By default, the destroyWindow property of newly created stacks is set to false.

Comments:

If a stack's destroyWindow property is set to true, closing the stack removes the data structure maintaining that window.

External commands and external functions are removed from memory only when the stack controlling them is removed from memory. When you install a new external, or replace it with a new version while debugging, Revolution cannot use it until you either quit Revolution and then reopen the stack, or close the stack (after setting its destroyWindow to true) and then reopen it.

Note: Despite its alarming name, the destroyWindow property does not destroy or damage a stack. It simply frees the memory used for its window.

dialogData

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer command, ask command, modal command, modeless command, How to create a custom dialog box

Summary

Specifies data to be passed to a dialog box.

Syntax

set the dialogData to string

Example Code

```
set the dialogData to "OK"  
put last item of the dialogData into buttonReceived
```

Comments

Use the dialogData property to pass data between a dialog box and the stack that opens the dialog box.

Value:

The dialogData is a string.

Comments:

The dialogData can be used for any data you wish to place in it. For example, you can use it within a modal dialog stack to hold the name of the button the user clicked, the state of other options that are set in the dialog box, the contents of text fields the user filled in, and any other information. The handler that showed the dialog box can then use the information in the dialogData. The calling handler can also place information—for example, default settings—into the dialogData property, and the modal dialog stack can use that information to set up the dialog box when it opens.

Since the dialogData property is global and can be set or accessed by any stack, this method generally creates simpler code than the alternatives (such as using global variables, or passing parameters between stacks).

Tip: The built-in "Answer Dialog" and "Ask Dialog" stacks in the file "license.rev", which are used by the ask and answer commands, use the dialogData property to pass data between the development environment and the dialog box. To see the scripts used for these stacks, enter one of the following statements into the message box:

edit script of card 1 of stack "Answer Dialog" of stack "Home"

edit script of card 1 of stack "Ask Dialog" of stack "Home"

disable

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantSelect property, disable menu command, disabled property, disabledIcon property, enable command, enabled property, hide command, traversalOn property

Summary

Dims a control so that it no longer responds to user actions.

Syntax

disable object

Example Code

```
disable field "Intro"
```

```
disable button 2 of card 7 of stack "Overall Controls"
```

Comments

Use the disable command to prevent an object from responding to mouse clicks or keyboard presses.

Parameters:

The object is any control in an open stack.

Comments:

The disable command is equivalent to setting the object's disabled property to true.

disable menu

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

disable command, disabled property, doMenu command, enable menu command, enabled property, About menus and the menu bar

Summary

Dims a menu or menu item so that it no longer responds to user actions.

Syntax

disable [menuItem itemNumber of] menu {menuName | menuNumber}
disable menuItem itemNumber of button

Example Code

```
disable menuItem 2 of menu "Styles"  
disable menu "Objects"  
disable menu 1
```

Comments

Use the disable menu command to dim a menu or menu item in the menu bar, preventing the menu item from being chosen.

Parameters:

The itemNumber is the number of a menu item, from top to bottom of the menu. The first menu item is numbered 1. (Horizontal lines count.)

The menuName is the name of a menu in the current menu bar.

The menuNumber is the number of a menu, from left to right.

Comments:

If a menuItem is specified, only that menu item is disabled; otherwise, the entire menu is disabled. The disable menu command inserts an open parenthesis (before the menu item's name; you can also manually disable a menu item by prepending an open parenthesis to its name.

When used to disable an entire menu at once, the disable menu command applies only to menus in the current menu bar. To disable a menu associated with a button, use the disable command.

On Mac OS systems, the Apple menu does not have a number and cannot be enabled or disabled. Menu numbering starts with the menu to the right of the Apple menu.

Since a Revolution menu bar is implemented as a group of buttons (one button per menu, with the menu items as lines in the button's text property), you can indicate a menu by specifying its button. Disabled menu items have an open parenthesis (at the beginning of the line for that menu item, so when used to disable a menu item, the disable menu command adds an open parenthesis at the beginning of the specified line.

disabled

property

Synonyms

Objects

control

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

cantSelect property, disable command, disabledIcon property, enabled property, traversalOn property

Summary

Specifies that an object is dimmed and does not respond to user action.

Syntax

set the disabled of object to {true | false}

Example Code

```
set the disabled of button "Destroy Everything" to true  
set the disabled of menu "View" to true
```

Comments

Use the disabled property to find out whether a control can respond to mouse clicks or keyboard presses.

Value:

The disabled of a control is true or false.

By default, the disabled property of newly created controls is false.

Comments:

Disabled buttons are drawn with gray borders and text to indicate that they cannot be clicked. Controls whose disabled property is true do not trigger messages such as mouseUp or mouseEnter, but they can respond to messages sent with the send command. If a disabled control is on top of an enabled control, the disabled control is transparent to clicks, and clicking sends the appropriate messages to the enabled control underneath.

A disabled control cannot receive the focus.

Setting a group's disabled property sets the disabled of each control in the group.

You can set the disabled property of a menu by specifying the menu's name or number. Menu items in disabled menus are drawn in gray text and cannot be chosen.

The disabled property of an object is the logical inverse of that object's enabled property. When the enabled is true, the disabled is false, and vice versa.

disabledIcon

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

disable command, disabled property, icon property, showIcon property, Why do icons disappear from a standalone application?

Summary

Specifies an image to display in a button when the button is disabled.

Syntax

set the disabledIcon of button to {imageID | imageName}

Example Code

```
set the disabledIcon of button "Boss Key" to "Innocent Spreadsheet"
```

Comments

Use the disabledIcon property to change a button's appearance when it is disabled.

Value:

The disabledIcon property is the ID or name of an image to use for an icon. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the disabledIcon property of newly created buttons is set to zero.

Comments:

When the button's enabled property is set to false, the disabledIcon is displayed within the button's rectangle.

If a button's showIcon property is false, the setting of its disabledIcon property has no effect.

diskSpace

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

defaultFolder property, hasMemory function, volumes function, About Revolution system requirements, Memory and Limits Reference

Summary

Returns the amount of free space on the disk that holds the defaultFolder.

Syntax

the diskSpace
diskSpace()

Example Code

```
the diskSpace  
put the diskSpace div (1024^2) into megabytesAvailable  
if the diskSpace < spaceNeeded then exit mouseUp
```

Comments

Use the diskSpace function to determine whether there is enough free space to complete an action (such as downloading data).

Value:

The diskSpace function returns an integer.

Comments:

The value returned by the diskSpace function is in bytes. To convert to kilobytes (K), divide the return value by 1024. To convert to megabytes (M), divide by 1024^2. To convert to gigabytes (G), divide by 1024^3:

```
function humanReadableDiskSpace theUnits  
    set the caseSensitive to false  
    switch char 1 of theUnits  
        case "k"  
            return the diskSpace div 1024
```

```
    break
  case "m"
    return the diskSpace div (1024^2)
    break
  case "g"
    return the diskSpace div (1024^3)
    break
  default
    return the diskSpace
    break
end switch
end humanReadableDiskSpace
```

div

operator

Synonyms

Objects

numeric

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

/ operator, mod operator, trunc function, Operator Precedence Reference

Summary

Divides one number by another and returns the integer part of the result.

Syntax

number div divisor

Example Code

```
4 div 2 -- evaluates to 2
11 div 4 -- evaluates to 2 (since 11/4 = 2.75)
arrayOfNumbers div 10
```

Comments

Use the div operator to do integer division.

Parameters:

The number is a number, or an expression that evaluates to a number, or an array containing only numbers.

The divisor is any non-zero number. If the number is an array, the divisor is either a non-zero number or an array containing only non-zero numbers.

Comments:

If the number to be divided is an array, each of the array elements must be a number. If an array is divided by a number, each element is divided by the number. If an array is divided by an array, both arrays must have the same number of elements and the same dimension, and each element in one array is divided by the corresponding element of the other array.

If an element of an array is empty, the div operator treats its contents as zero.

The expression dividend div divisor is equivalent to trunc(dividend/divisor).

If dividend can be divided evenly into divisor, the expression `dividend div divisor` is equal to `dividend/divisor`.

Attempting to divide by zero causes an execution error.

Note: While using non-integer number and divisor usually produces sensible results, mathematically, integer division is generally defined as a function over the integers, and the results using non-integers may not consistently be what you expect.

Changes to Transcript:

The option to divide arrays was introduced in version 1.1. In previous versions, only single numbers could be used with the `div` operator.

divide

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

* operator, add command, multiply command, numberFormat property, subtract command

Summary

Divides a container by a number and places the resulting value in the container.

Syntax

divide [chunk of] container by number
divide arrayContainer by {number | array}

Example Code

```
divide it by 50  
divide field "Total Income" by 12  
divide line x of field "Expenses" by percentageRate  
divide yearlyTotals by 52
```

Comments

Use the divide command to divide a container or a portion of a container by a number, or to divide two arrays containing numbers.

Parameters:

The chunk is a chunk expression specifying a portion of the container.

The container is a field, button, or variable, or the message box.

The number is any expression that evaluates to a number.

Comments:

The contents of the container (or the chunk of the container) must be a number or an expression that evaluates to a number.

If an arrayContainer is divided by a number, each element is divided by the number. If an arrayContainer is divided by an array, both arrays must have the same number of elements and the same dimension, and each element in the arrayContainer is divided by the corresponding element of the array.

If the container or an element of the arrayContainer is empty, the divide command treats its contents as zero.

If container is a field or button, the format of the result is determined by the numberFormat property.

Attempting to divide by zero causes an execution error.

Changes to Transcript:

The divide arrayContainer form was introduced in version 1.1. In previous versions, only single numbers could be used with the divide command.

DNSServers

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

hostAddress function, hostName function, hostNameToAddress function, peerAddress function

Summary

Returns a list of the DNS servers listed in the system's TCP/IP network configuration.

Syntax

the DNSServers

DNSServers()

Example Code

```
put the DNSServers into myDNSList
```

Comments

Use the DNSServers function to find the IP address of the servers that will be used to translate IP addresses into domain names.

Value:

The DNSServers function returns a list of IP addresses, one per line.

Cross-platform note: On Mac OS systems, the DNSServers function returns only the first IP address.

Comments:

When an application requests contact with a system specified by its domain name, the domain name must first be translated into an IP address. A DNS server performs this translation function. In your system's TCP/IP configuration, one or more DNS servers are specified by their IP addresses, and these servers are queried when a domain name needs to be translated to an IP address. (Typically, one or more DNS servers will be configured for an ISP or corporate network; name service is usually centralized.)

Since the DNSServers function reads the value set in the network configuration, it is available even if the system is not currently connected to the Internet.

do

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

How to schedule a future event, alternateLanguages function, call command, debugDo command, params function, pass control structure, revMacFromUnixPath function, revUnixFromMacPath function, send command, value function

Summary

Executes a list of statements.

Syntax

do statementList [as OSALanguageName]

Example Code

```
do "go next card"
do "put" && x && "into myVar" & x -- might become "put 3 into myVar3"
do "select" && line 3 of field "Objects"
do field "Statements" as AppleScript
```

Comments

Use the do command to execute statements in a container, or to execute a statement that consists partly of a literal string and partly of a container or the return value from a function.

Parameters:

The statementList is a Transcript statement, a container with one or more statements, or a string that evaluates to a statement.

On Mac OS and OS X systems, the OSALanguageName is a script language (such as AppleScript) supported under the Open Scripting Architecture. The available languages are returned by the alternateLanguages function. If you specify an OSALanguageName, the statementList must be written in the specified language.

Comments:

Using the do command is slower than directly executing the commands, because each statement must be compiled every time the do command is executed.

If you use the `do as OSALanguageName` form, any result returned by the script language is placed in the result.

Important! If you use the `do as OSALanguageName` form, any file paths you use in the `statementList` must be formatted as Mac OS standard paths. The same caveat applies when executing an OSA `statementList` that returns a file path: OSA languages return standard Mac OS file paths, and these must be converted to Revolution file paths before you can use them in a Transcript statement. To convert between Mac OS paths and Revolution paths, use the `revMacFromUnixPath` and `revUnixFromMacPath` functions.

When using a standalone application created by Revolution, you can include up to ten statements in the `statementList`. This limit is set by line 2 of the `scriptLimits` function. This limitation does not apply when using the development environment.

Note: The ten-statement limit in standalones applies only if the `statementList` is written in Transcript. If you specify an OSA language, this limit does not apply.

If you attempt to specify an `OSALanguageName` on a Unix or Windows system, the `do` command is not executed, and the result is set to "alternate language not found".

Changes to Transcript:

The `OSALanguageName` option was introduced in version 1.1. In previous versions, it was not possible to include AppleScript or other OSA languages in a Transcript handler.

doMenu

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

copy command, cut command, disable menu command, enable menu command, paste command, quit command, undo command, How to simulate a menu choice

Summary

Simulates a choice from a menu.

Syntax

doMenu menuItem [of menu menuName]

Example Code

```
doMenu "New Card" of menu "Object"  
doMenu (line 10 of thingsToDo)
```

Comments

Use the doMenu command to choose a menu item.

Parameters:

The menuItem is the name of a menu item.

The menuName is the name of the menu in which the menuItem appears.

Comments:

The doMenu command is not implemented for all menu items. This command is included in Transcript for compatibility with imported HyperCard stacks.

done
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

beep command, movie function, play command, sound function

Summary

Equivalent to the string "done".

Syntax

Example Code

```
wait until the sound is done  
if the sound is done then go next
```

Comments

The done constant is returned by the sound and movie functions to indicate that no audio clip or video clip is currently playing.

dontDither

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

colorMap property, screenDepth function, screenGamma property

Summary

Specifies whether 24-bit images are dithered on displays with fewer colors.

Syntax

set the dontDither of image to {true | false}

Example Code

```
set the dontDither of image "Photo" to false
set the dontDither of image "Line Art" to true
```

Comments

Use the dontDither property to control how bitmapped images such as JPEGs or PNGs behave when the stack is used on a display whose bit depth is 8 bits (256 colors) or fewer.

Value:

The dontDither of an image is true or false.

By default, the dontDither property of newly created images is set to false.

Comments:

If the dontDither of an image is set to true, then when that image is displayed on an 8-bit screen, colors that are not part of the current color table are displayed as a solid block of the closest available color. This may create visible banding on images with continuous gradations of color.

If the dontDither of an image is false, colors that are not part of the current color table are displayed as a dithered pattern made up of two or more close colors, in order to approximate the exact color. This is preferable for images such as photographs that include continuous gradations of color, but may produce speckling on images with large blocks of solid color.

The setting of the dontDither property has no effect on 24-bit or 16-bit screens.

dontRefresh

property

Synonyms

Objects

videoClip

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

play command, screenColors function, templateVideoClip keyword, videoClipPlayer property

Summary

Specifies whether the last frame of a video clip remains visible or is cleared at the end of the movie.

Syntax

set the dontRefresh of videoClip to {true | false}

Example Code

```
set the dontRefresh of videoclip "Sample Movie" to true
```

Comments

Use the dontRefresh property to leave the final frame of a movie visible.

Value:

The dontRefresh of a video clip is true or false.

By default, the dontRefresh property of newly created video clips is set to false.

Comments:

If a video clip's dontRefresh property is true, the last frame of the movie remains on the screen after the movie is finished playing, until that portion of the screen is redrawn.

If the dontRefresh is false, the last frame disappears at the end of the movie.

Note: The color table used by the movie does not persist after the movie finishes playing. This means that if the screenColors is 256 or fewer, the colors in the last frame may shift when the movie ends. To prevent this problem, you can set the colorMap property to the set of colors used by the movie.

dontResize

property

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

drag command, points property, style property

Summary

Specifies whether an irregular polygon graphic is resized when its vertexes are changed.

Syntax

set the dontResize of graphic to {true | false}

Example Code

```
set the dontResize of graphic 12 to true
```

Comments

Use the dontResize property to automatically resize and reposition a graphic when you set its points property.

Value:

The dontResize of a graphic is true or false.

By default, the dontResize property of newly created graphics is set to false.

Comments:

You can reshape a graphic whose style property is set to polygon by changing its points or relativePoints property.

If you change the graphic's points or relativePoints property, and the graphic's dontResize property is set to false, the graphic is resized and repositioned if necessary to accommodate the new shape.

If you change the points or relativePoints and the dontResize is false, the graphic is not resized to fit the new set of vertexes. This may result in parts of the shape being cut off at the boundaries of the graphic.

dontSearch

property

Synonyms

Objects

field, group, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

find command, lockText property, sharedText property

Summary

Specifies whether the find command skips a field, group, or card.

Syntax

set the dontSearch of field to {true | false}

Example Code

```
set the dontSearch of field "Label" to true
```

Comments

Use the dontSearch property to skip certain fields (such as label fields that contain static text), or to skip all the fields in a specified group or card, when searching with the find command.

Value:

The dontSearch of a field, group, or card is true or false.

By default, the dontSearch property of newly created fields, groups, and cards is set to false.

dontUseNS

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

answer file command, ask file command, save command, systemFileSelector property

Summary

Specifies whether Revolution uses old-style file dialog boxes or Navigation Services file dialog boxes on Mac OS systems.

Syntax

set the dontUseNS to {true | false}

Example Code

```
set the dontUseNS to true
set the dontUseNS to not the hilited of button "Navigation Services"
```

Comments

Use the dontUseNS property to change the behavior and appearance of file dialog boxes.

Value:

The dontUseNS is true or false.

By default, the dontUseNS property is set to false if the system version is 8.6 or later, true otherwise.

Comments:

On Mac OS systems with Navigation Services installed, applications can use a new implementation of the standard file dialog boxes. Navigation Services dialogs have a title bar and are movable. By default, Revolution supports Navigation Services and uses the new dialog boxes. To disable this feature and use the old-style modal dialog boxes for file actions, set the dontUseNS property to true.

The setting of this property affects all file dialog boxes used in Revolution, including the dialogs displayed by the ask file, answer file, and answer folder commands, as well as dialog boxes displayed by menu items such as "Save" and "Open Stack".

Caution! Navigation Services version 1.1.2 is part of Mac OS version 8.6 and later. Earlier versions may optionally be installed on systems with Mac OS versions 8.1 and 8.5. Versions of Navigation Services earlier than 1.1.2 may not display file dialogs correctly in Revolution.

dontUseQT

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer effect command, answer record command, dontUseQTEffects property, play command, QTVersion function, record sound command, videoClipPlayer property, visual effect command, How to find out whether QuickTime is available

Summary

Disables the ability to use QuickTime.

Syntax

set the dontUseQT to {true | false}

Example Code

```
set the dontUseQT to true
if the platform is "Win32" then set the dontUseQT to true
```

Comments

Use the dontUseQT property to test operation of a stack using the built-in MCI video on Windows systems, or to use built-in visual effects rather than QuickTime effects.

Value:

The dontUseQT is true or false.

By default, the dontUseQT property is set to false.

Comments:

If you have already used QuickTime during the current session, setting the dontUseQT property to true has no effect, because the code for Revolution to use QuickTime is already loaded into memory and will continue to be used until you quit the application.

The following actions cause QuickTime to be loaded into memory:

- Displaying a player object
- Playing an audio clip or video clip
- Recording sound using the record sound command

- Using the recordFormats function or the answer record command
- Using a visual effect with visual effect, show, hide, or unlock screen
- Using the answer effect command or QTEffects function
- Using the QTVersion function

Tip: It may take Revolution a second or two to load the code needed to use QuickTime, depending on the machine speed. Since this code is only loaded into memory once per session, you can speed up the first occurrence of an action listed above by calling the QTVersion function during otherwise dead time—for example, during startup of your application—to preload QuickTime.

The setting of this property has no effect on Unix systems.

This property is of limited usefulness on Mac OS systems, since they normally use QuickTime for playing movies. Setting the dontUseQT to true on a Mac OS system prevents it from playing movies and sounds.

dontUseQTEffects

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer effect command, dontUseQT property, QTVersion function, visual effect command, How to find out whether QuickTime is available

Summary

Disables QuickTime visual effects.

Syntax

set the dontUseQTEffects to {true | false}

Example Code

```
set the dontUseQTEffects to false
if not the wantEffects of me then set the dontUseQTEffects of me to true
```

Comments

Use the dontUseQTEffects property on slower machines for better visual results on slower machines when using built-in effects.

Value:

The dontUseQTEffects is true or false.

By default, the dontUseQTEffects property is set to false.

Comments:

Visual effects are used with the visual effect, hide, show, and unlock screen commands. By default, Revolution uses QuickTime to produce these transition effects. However, slower machines may not be powerful enough to produce good results when using QuickTime for transition effects. Typically, this problem appears as "stuttering" transitions or transitions with strange visual artifacts such as diagonal lines moving across the stack window.

If the dontUseQTEffects property is set to true, Revolution uses its own built-in routines for visual effects, rather than QuickTime. The built-in routines, while less versatile, require less computing power and produce a good appearance on all machines.

Note: If the `dontUseQTEffects` property is true, visual effects generated by the `answer effect` command cannot be used.

The setting of this property has no effect on Unix systems.

dontWrap

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

formattedText property, formattedWidth property, hScrollbar property, width property, Recipe for truncating text to a pixel width

Summary

Specifies whether the text in a field word-wraps when it reaches the field's right edge.

Syntax

set the dontWrap of field to {true | false}

Example Code

```
set the dontWrap of field "Label" to true
```

Comments

Use the dontWrap property to control the appearance and behavior of a field.

Value:

The dontWrap of a field is true or false.

By default, the dontWrap property of newly created fields is set to false.

Comments:

If a field's dontWrap property is false, the text in the field wraps to the next line automatically when it reaches the right edge of the field, so all the text is visible in the field's width.

If the dontWrap is true, lines that are too long to fit in the width of the field disappear off the right edge of the field. (The excess text is still in the fieldóit just can't be seen.) To edit long lines in a field whose dontWrap is true, set the field's hScrollbar to true to display a horizontal scrollbar.

doubleClickDelta

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

doubleClickInterval property, mouseDoubleDown message, mouseDoubleUp message

Summary

Specifies the distance the mouse can move between clicks during a double click.

Syntax

set the doubleClickDelta to numberOfPixels

Example Code

```
set the doubleClickDelta to 10
```

Comments

Use the doubleClickDelta property to fine-tune double-clicking.

Value:

The doubleClickDelta is an integer between zero and 32767.

By default, the doubleClickDelta is set to 4 pixels.

Comments:

If the mouse moves more than the specified number of pixels in any direction between the first and second click, no mouseDoubleDown or mouseDoubleUp message is sent to the clicked object. If the doubleClickDelta is set to zero, the mouse may not move at all during a double click.

doubleClickInterval

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

doubleClickDelta property, mouseDoubleClickDown message, mouseDoubleClickUp message

Summary

Specifies how long the delay can be between two clicks for them to be considered a double click.

Syntax

set the doubleClickInterval to timeInterval

Example Code

```
set the doubleClickInterval to 100 -- 1/10 second  
set the doubleClickInterval to 250 -- 1/4 second
```

Comments

Use the doubleClickInterval property to fine-tune double-clicking.

Value:

The doubleClickInterval is an integer between zero and 65535. The value of the doubleClickInterval is expressed in milliseconds.

Comments:

Increase the doubleClickInterval to make the mouse less sensitive to slow double clicks; decrease it to make it less likely that two separate clicks will be mistaken for a double click.

If more time than the doubleClickInterval elapses between the first and second click, no mouseDoubleClickDown or mouseDoubleClickUp message is sent to the clicked object.

On Mac OS systems, the doubleClickInterval is set to the current setting (in the Mouse control panel) when Revolution starts up. Normally, you should not change the doubleClickInterval setting, since doing so may discombobulate the user.

On Windows systems, the doubleClickInterval property is read-only and cannot be set.

down
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

altKey function, commandKey function, controlKey function, left constant, mouse function, optionKey function, right constant, shiftKey function, up constant

Summary

Equivalent to the string "down".

Syntax

Example Code

```
wait while the mouse is down  
if the altKey is down then showHelpText
```

Comments

Use the down constant to indicate the state of a mouse button or modifier key, or in an arrowKey handler.

drag

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

click command, dragSpeed property, mouseDown message, mouseMove message, mouseUp message, move command

Summary

Simulates a mouse click and drag.

Syntax

drag [button number] from start to end [with key [,key [key]]]

Example Code

```
drag from "120,89" to "180,2" with commandKey  
drag from the clickLoc to the loc of button "Trash"  
drag button 2 from "500,200" to savedLoc with shiftKey,optionKey
```

Comments

Use the drag command to paint with the paint tools.

Parameters:

The number is the number of a mouse button. If you don't specify a number, button 1 is used.

The start is any expression that evaluates to a point—a vertical and horizontal distance from the top left of the current stack, separated by a comma.

The end is any expression that evaluates to a point.

The key is one of `commandKey`, `controlKey`, `optionKey`, or `shiftKey`. You can specify up to three keys, separated by commas. (On Windows and Unix, `commandKey` indicates the Control key.)

Comments:

You can use the drag command to move a control, but it is more efficient to use the move command instead.

The `dragSpeed` property determines how fast the drag motion is.

dragData

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

clipboardData property, dragDrop message, dragStart message, htmlText property, RTFText property, unicodeText property, How to start a drag and drop, How to respond to a drag and drop, Recipe for dragging a background color onto an object

Summary

Specifies what data is being dragged during a drag and drop.

Syntax

set the dragData to textToDrag

set the dragData[dataType] to dataToDrag

Example Code

```
set the dragData["html"] to the htmlText of field 1
get URL the dragData["text"]
```

Comments

Use the dragData property to find out what is being dragged or to change the data being dragged during the drag.

Value:

The dragData is an array with one or more of the following elements:

- text The plain text being dragged.
- HTML The styled text being dragged, in the same format as the htmlText
- RTF The styled text being dragged, in the same format as the RTFText
- Unicode The text being dragged, in the same format as the unicodeText
- image The data of an image (in PNG format)
- files The name and location of the file or files being dragged, one per line

Comments:

Which elements are present in the dragData array may depend on what type of data is being dragged. It is possible for more than one element in the dragData array to be populated. Different drop destinations

can each use the data type they need. (For example, a list field's dragDrop handler might use the `dragData["text"]` to ignore the styles, , while an ordinary field's dragDrop handler might use the `dragData["HTML"]` to retain the styles.

When dragging text in an unlocked field, Revolution automatically sets the `dragData["text"]` to the plain text being dragged, and sets the `dragData["HTML"]`, the `dragData["RTF"]`, and the `dragData["Unicode"]` appropriately. To override this behavior, set the `dragData` in the field's `dragStart` handler.

To enable a drag from another object type, set the `dragData` to an appropriate value in the object's `mouseDown` handler. This starts the drag. The destination object's `dragDrop` handler should check the value of the `dragData` and do whatever is required with it.

If you don't specify a `dataType`, the `dragData` property reports or sets the `dragData["text"]`.

Note: It is necessary to use double quotes around the keys of the `dragData` array because some of the key names (such as "files") are Transcript reserved words. Quoting array indexes (other than integers) is a good practice anyway, but in the case of this property, it's necessary to avoid script errors.

dragDestination

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

dragDrop message, dragEnd message, dragSource function, target function

Summary

Returns the long ID of the object that dragged data was dropped on.

Syntax

the dragDestination

dragDestination()

Example Code

```
the dragDestination
```

```
put the rect of the dragDestination into whereWeDragged
```

Comments

Use the dragDestination function to determine which object received a drag and drop.

Value:

The dragDestination function returns the long ID property of the object.

Comments:

The dragDestination function only returns a value while a dragDrop handler is being executed: that is, you can use it in a dragDrop handler, or in a handler that's called by a dragDrop handler. If called from another handler, the dragDestination function returns empty.

dragDrop

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

acceptDrop property, dragEnd message, dragDestination function, dragLeave message, dragSource function, dragStart message, dropChunk function, mouseUp message, How to copy dragged and dropped text, How to prevent dragging and dropping to a field, How to respond to a drag and drop, Recipe for dragging a background color onto an object

Summary

Sent to the object where data was dropped when a drag and drop finishes.

Syntax

dragDrop

Example Code

```
on dragDrop -- check whether a file is being dropped
  if the dragData["files"] is empty then beep 2
  pass dragDrop
end dragDrop
```

Comments

Handle the dragDrop message to perform an action when the user drops data, or trap the message to prevent text from being dropped.

Comments:

The dragDrop message is sent to the control where data is being dragged.

Important! If the acceptDrop property is set to false at the time the drop occurs, no dragDrop message is sent.

Revolution automatically handles the mechanics of dragging and dropping text between and within unlocked fields. To support this type of drag and drop operation, you don't need to do any scripting: the text is dropped into the field automatically when Revolution receives the dragDrop message.

To prevent an unlocked field from accepting a drag and drop, trap the dragDrop message. If you don't want to allow a particular field to accept text drops, place a dragDrop handler in the field that does not contain a pass control structure:

```
on dragDrop -- in script of field or one of its owners
  -- do nothing, but trap the message
end dragDrop
```

On the other hand, if you want to perform some action when text is dropped into an unlocked field, you need to pass the dragDrop message once you're done with it in order to allow the drop to take place:

```
on dragDrop
  set the cursor to 9023
  pass dragDrop -- needed for drop to occur
end dragDrop
```

To accept drops to a locked field or a control other than a field, handle the dragDrop message, using the dragData property to determine what data is being dragged. For these objects, your dragDrop handler must get the dragged data and put it into the object (or perform whatever action you want to do on a drop); the behavior is not automated as it is for unlocked fields.

dragEnd

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

acceptDrop property, dragDestination function, dragDrop message, dragLeave message, dragSource function, dragStart message, mouseRelease message, How to copy dragged and dropped text, Recipe for dragging a background color onto an object

Summary

Sent to the object a drag and drop started from, when the data is dropped.

Syntax

dragEnd

Example Code

```
on dragEnd -- remove data being dragged
    delete the dragSource
end dragEnd
```

Comments

Handle the dragEnd message to perform an action when the user drops data, or trap the message to prevent text from being deleted from a field when it's dragged to another field.

Comments:

The dragEnd message is sent to the control that the drag started from. If the data is not dropped onto a control that accepts it, the dragEnd message is still sent.

Revolution automatically handles the mechanics of dragging and dropping text between and within unlocked fields. To support this type of drag and drop operation, you don't need to do any scripting: the text is deleted from the source field automatically when Revolution receives the dragEnd message.

To prevent the dragged text from being automatically deleted from the source field when it's dropped onto another unlocked field, trap the dragEnd message. If you don't want text from a particular field to be deleted after dragging, place a dragEnd handler in the field that does not contain a pass control structure:

```
on dragEnd -- in script of field or one of its owners
  -- do nothing, but trap the message
end dragEnd
```

On the other hand, if you want to perform some action when text is dragged from an unlocked field and then dropped, you need to pass the dragEnd message once you're done with it in order to allow the deletion to take place:

```
on dragEnd
  set the textColor of me to "gray"
  pass dragEnd -- needed for text removal to occur
end dragEnd
```

To make changes to the original data in a locked field or a control other than a field when a drop is completed, handle the dragEnd message. For these objects, if you want the dragged data to be removed (or changed), your dragEnd handler must delete the dragged text from the source object (or perform whatever action you want to do on a drop); the behavior is not automated as it is for unlocked fields.

dragEnter

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

dragLeave message, dragMove message, dragStart message, mouseLoc function, mouseEnter message, How to prevent dragging and dropping to a field, Recipe for dragging a background color onto an object

Summary

Sent when the mouse pointer moves into an object during a drag and drop.

Syntax

dragEnter

Example Code

```
on dragEnter -- show a green outline around the drop target
    set the borderColor of the target to "green"
end dragEnter
```

Comments

Handle the dragEnter message to perform an action (for example, display a special cursor or set the acceptDrop property) when the mouse pointer enters an object.

Comments:

The dragEnter message is sent only when a drag and drop operation is in progress.

If two controls overlap, a dragEnter message is sent whenever the mouse pointer crosses into a visible portion of a control. The control on the bottom receives a dragEnter message only when the mouse pointer enters part of the control that can be seen. A control that is completely hidden by another control on top of it will never receive a dragEnter message.

Tip: Use the mouseLoc function to determine where the mouse pointer is. Use the target function to determine which control the mouse pointer has entered.

draggable

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

decorations property, title keyword, zoomBox property

Summary

Shows a stack's title bar.

Syntax

set the draggable of stack to {true | false}

Example Code

Comments

The draggable property is included in Transcript for compatibility with imported SuperCard projects.

In SuperCard, the draggable property determines whether a window can be dragged by its title bar.

In Revolution, setting this property sets the stack's decorations property to "title". Setting the stack's draggable back to false does not change the stack's decorations.

Setting the draggable property causes the stack window to flash briefly.

dragLeave

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

dragEnter message, dragMove message, dragStart message, mouseLoc function, mouseLeave message, How to prevent dragging and dropping to a field

Summary

Sent when the mouse pointer moves out of an object during a drag and drop.

Syntax

dragLeave

Example Code

```
on dragLeave -- remove any outline around the drop no-longer-target
    set the borderColor of the target to empty
end dragLeave
```

Comments

Handle the dragLeave message to update a control when the mouse pointer moves outside it during drag and drop.

Comments:

The dragLeave message is sent only when a drag and drop operation is in progress.

If two controls overlap, a dragLeave message is sent whenever the mouse pointer crosses outside a visible portion of a control. The control on the bottom receives a dragLeave message only when the mouse pointer leaves part of the control that can be seen. A control that is completely hidden by another control on top of it will never receive a dragLeave message.

You can use the dragEnter and dragLeave messages to display a visual cue that shows which control will receive the drop if the user releases the mouse button. For example, the following handlers turn the border of a "draggable" object green while the mouse is over it:

```
on dragEnter
    set the borderColor of the target to "green"
```

```
end dragEnter
```

```
on dragLeave
```

```
    set the borderColor of the target to empty
```

```
end dragLeave
```


dragMove

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

dragEnter message, dragLeave message, dragStart message, mouseMove function, mouseWithin message

Summary

Sent when the user moves the mouse during a drag and drop.

Syntax

dragMove

Example Code

```
on dragMove -- in a field script
  -- set the cursor so it shows you can only drop onto a link:
  if the textStyle of the mouseChunk contains "link"
    then set the cursor to the ID of image "Drop Here"
  else set the cursor to the ID of image "Dont Drop"
end dragMove
```

Comments

Handle the dragMove message if you want to perform some action (such as updating a status display) when the user moves the mouse while dragging, or if you want to keep continuous track of the mouse pointer's position during a drag and drop.

Comments:

The dragMove message is sent to the control the mouse pointer is over, or to the card if no control is under the mouse pointer.

dragSource

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

dragDestination function, dragDrop message, dragEnd message, target function

Summary

Returns the long ID of the object that dragged data was dragged from.

Syntax

the dragSource

dragSource()

Example Code

```
the dragSource
```

```
if the short name of the dragSource is empty then beep
```

Comments

Use the dragSource function to find out which object, and which object type, is being dragged from.

Value:

The dragSource function returns the long ID property of the object.

If the drag came from another application, the dragSource returns an application identifier, or empty if the source cannot be determined.

Comments:

The dragSource function only returns a value while a drag and drop is in progress. Otherwise, the dragSource function returns empty.

dragSpeed

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

click command, drag command, moveSpeed property

Summary

Specifies the speed of the drag command.

Syntax

set the dragSpeed to pixelsPerSecond

Example Code

```
set the dragSpeed to 10
```

Comments

Use the dragSpeed property to control the speed of the drag command.

Value:

The dragSpeed is an integer between zero and 65535.

By default, the dragSpeed property is set to zero. It is reset to zero when no handlers are executing.

Comments:

The dragSpeed specifies how many pixels per second the drag command moves.

The setting of the dragSpeed affects the smoothness of drags made with some of the paint tools (such as the Spray Can tool) whose action is speed-dependent.

If the dragSpeed is zero, the drag command moves as fast as possible. Dragging with the dragSpeed set to zero is equivalent to clicking the start location and immediately releasing the mouse button at the end location. This means you cannot drag shapes other than a straight line if the dragSpeed is zero.

dragStart

message

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

click command, dragData property, dragDrop message, dragEnd message, dragEnter message, dragSource function, How to copy dragged and dropped text

Summary

Sent when the user begins to drag selected text from a field.

Syntax

dragStart

Example Code

```
on dragStart -- in a field script
  -- color the text that's being dragged:
  set the textColor of the selectedChunk to "#333399"
  pass dragStart
end dragStart
```

Comments

Handle the dragStart message to perform an action (such as changing the cursor) when the user starts dragging, or trap the message to prevent text from being dragged.

Comments:

The dragStart message is sent to the field where the drag is starting.

If the dragStart handler does not pass the message or send it to a further object in the message path, the drag and drop does not occur. Passing the message allows the drag and drop to start.

Revolution automatically handles the mechanics of dragging and dropping text between and within unlocked fields. To support this type of drag and drop operation, you don't need to do any scripting: the drag begins automatically when Revolution receives the dragStart message.

If you don't want to allow dragging text from a particular field, you should place a dragStart handler in the field that does not contain a pass control structure:

```
on dragStart -- in script of field or one of its owners
  -- do nothing, but trap the message
end dragStart
```

On the other hand, if you want to perform some action at the start of a drag, you need to pass the dragStart message once you're done with it in order to allow the drag to start:

```
on dragStart -- in script of field or one of its owners
  set the cursor to 9023
  pass dragStart -- needed for drag to occur
end dragStart
```

drawer

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

go command, mode property, palette command, topLevel command, About windows, palettes, and dialog boxes, How to open and close a drawer

Summary

Displays a stack as a drawer of another stack.

Syntax

drawer stack {[at] left | bottom | right} [{of | in} parentStack]

[aligned [to] {left | right | top | bottom | center}]

Example Code

```
drawer stack "Recent Transactions" at right
drawer stack "Tips" at left of stack "Main" aligned to top
drawer me at bottom aligned to left
drawer stack "Help List" at right of this stack
drawer stack "Connections" at right aligned to center
```

Comments

Use the drawer command to slide a stack out of one edge of another stack.

Parameters:

The stack is any stack reference.

The parentStack is any reference to a stack in an open window.

Comments:

The drawer command opens the stack as a drawer of the specified parentStack. (The stack's rectangle and location properties are ignored.)

You can open a drawer at the left, bottom, or right edge of the parentStack. If you don't specify an edge, the drawer opens at the left if there is room for it to fully open. Otherwise, it opens at the right.

You can align a drawer to the top, bottom, or center (of the parentStack's left or right edge) or to the left, right, or center (of the bottom edge). If you don't specify an alignment, the drawer is aligned to the center of the specified edge. To let the user access multiple drawers on one side of a window, align them to different locations on that side:

```
drawer "Basic Tools" at left of this stack aligned to top  
drawer "Colors" at left of this stack aligned to bottom
```

Cross-platform note: On Mac OS, Unix, and Windows systems, drawers are not supported, so the drawer command opens the stack as a palette instead. The palette uses the current rectangle of the stack and does not resize or move it.

To slide the drawer back in, either close the stack or hide it:

```
close stack "Colors"  
hide stack "Colors"
```

If the stack is already open, the drawer command closes the stack and reopens it as a drawer, so closeStack and openStack, closeCard and openCard, and (if applicable) closeBackground and openBackground messages are sent to the current card as a result of executing this command. Use the lock messages command before executing modal if you want to prevent the close messages from being sent; the open messages are sent regardless of the setting of the lockMessages property.

driverNames

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

close driver command, open driver command, read from driver command, serialControlString property, write to driver command

Summary

Returns a list of available serial drivers.

Syntax

the driverNames

driverNames()

Example Code

```
put firstItems(the driverNames) into menu "Choose a driver:"
```

Comments

Use the driverNames function to find out which serial drivers can be used with the open driver, read from driver, write to driver, and close driver commands.

Value:

The driverNames function returns a list of drivers, one per line. Each line consists of three items, separated by commas:

1. The name of the driver.
2. The name and location of the TTY device file for the driver.
3. The name and location of the callout device (CU) file for the driver.

Comments:

Usually, you use the callout file nameóthe third item in the line for the driver you wantówith the open driver, read from driver, write to driver, and close driver commands.

If the driverNames function is called on a Mac OS, Unix, or Windows system, it returns empty and the result returns "not supported".

dropChunk

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

clickChunk function, dragDestination function, dragDrop message, mouseChunk function

Summary

Returns a chunk expression describing the location in the field where data was dropped.

Syntax

the dropChunk

dropChunk()

Example Code

```
the dropChunk  
select the dropChunk
```

Comments

Use the dropChunk function within a dragDrop handler to find out exactly where text was dropped into a field.

Value:

The dropChunk function returns a chunk expression of the form char followingChar to precedingChar of field fieldNumber. The followingChar is always one less than the precedingChar.

Comments:

The dropChunk function only returns a value while a dragDrop handler is being executed: that is, you can use it in a dragDrop handler or in a handler that's called by a dragDrop handler. If called from another handler, the dropChunk function returns empty.

The dropChunk function designates an insertion point rather than a run of characters. For example, if you drop text between characters 3 and 4 of field number 1, the dropChunk function returns "char 4 to 3 of field 1".

dropper

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

choose command, mouseColor function, penColor property, tool function

Summary

Designates the paint tool used to pick up a color in an image and make it the paint color.

Syntax

Example Code

```
choose dropper tool
```

Comments

Use the dropper keyword to change the penColor.

Comments:

When using the Dropper tool, the cursor is an eyedropper shape. This tool is used to pick up a color from the screen to use in further painting. The penColor is changed to the color of the pixel you clicked.

You can click anywhere in a stack window to pick up a color.

duration

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentTime property, endTime property, play command, playRate property, startTime property, timeScale property

Summary

Specifies the length of a sound or movie.

Syntax

get the duration of player

Example Code

```
put (the duration of me/the timeScale of me) into totalSeconds
```

Comments

Use the duration property to find out how long a movie or sound takes to play.

Value:

The duration of a player is an integer.

This property is read-only and cannot be set.

Comments:

The duration of a player is the number of intervals in the movie or sound contained in the player. (The number of intervals per second is specified by the player's timeScale property.)

dynamicPaths

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundBehavior property, HCAddressing property, HCStack property, pass control structure, send command, About Revolution for HyperCard developers

Summary

Specifies whether the message path includes the current card and its owners.

Syntax

set the dynamicPaths of stack to {true | false}

Example Code

```
set the dynamicPaths of this stack to true
```

Comments

Use the dynamicPaths property for compatibility with imported HyperCard stacks that rely on HyperCard's dynamic path behavior.

Value:

The dynamicPaths of a stack is true or false.

By default, the dynamicPaths property of newly created stacks is set to false.

Comments:

If a stack's dynamicPaths property is true, when a handler in that stack uses the go or find command to go to a card other than the original card, that destination card's message path is inserted into the message path as long as the handler is on that card.

If the dynamicPaths is false, the message path does not change even when a handler visits another card.

The dynamicPaths property of stacks imported from HyperCard is set to true by default.

each

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

by keyword, item keyword, line keyword, sort container command, Recipe for listing the unique words in a string, Recipe for sorting a list of titles

Summary

Used with the sort container command to specify that the sort key consists of the specified chunk of each line or item. Also used with the for keyword in a repeat control structure to specify that the repeat loop should be executed for each chunk of the specified string or each element of the specified array.

Syntax

Example Code

```
sort lines of field "Address" by word 2 of each
repeat for each character thisChar in field "Input"
```

Comments

Use the each keyword to sort a container by a portion of each item or line, rather than by the whole item or line, or to iterate over each chunk in a string or each element in an array.

Comments:

In a sort container command, the expression before "of each" must be a chunk expression. The chunk expression describes a portion of each item or line in the container to be sorted.

If a chunk is nonexistent for a particular line or item—for example, if "by char 5 of each" is specified and some lines only have four characters—the chunk is treated as though it were empty.

Changes to Transcript:

The ability to specify each element of an array in a repeat control structure was introduced in version 1.1. In previous versions, you could use the each keyword to iterate over all chunks in a string, but not over all elements in an array.

edit

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

editScript message, licensed function, script property, Shortcut to edit the stack script, Shortcut to edit the card script, Shortcut to edit a control's script, Shortcut to edit the selected control's script

Summary

Opens the script of an object in the development environment.

Syntax

edit [the] script of object

Example Code

```
edit the script of the mouseStack
edit the script of this card
edit the script of player "Show Me"
```

Comments

Use the edit command in the development environment to open the Revolution script editor with the script of the specified object.

Parameters:

The object is any object reference.

Comments:

The edit command is equivalent to selecting the object and choosing Object menu Object Script. If the script editor is not open, the edit command opens a window that shows the script.

editBackground

property

Synonyms

editBkgnd, editBg

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

groupNames property, relayerGroupedControls property, start editing command, stop editing command, About groups and backgrounds

Summary

Specifies that any objects created should be added to the background.

Syntax

set the editBackground of stack to {true | false}

Example Code

```
set the editBackground of this stack to true
```

Comments

Use the editBackground property to edit the first group on the current card, or to find out whether the stack is in group-editing mode.

Value:

The editBackground of a stack is true or false.

By default, the editBackground property of newly created stacks is set to false.

Comments:

If the editBackground property is set to true, any newly created objects become part of the group being edited. If the editBackground is false, newly created controls are created as card controls and only appear on the current card.

If the current card has more than one group, the group whose number is lowest is edited when you set the editBackground of the stack to true.

You can use the start editing command to edit a specific group. Setting the editBackground property to false is equivalent to the stop editing command.

By default, the `editBackground` is set to `false`.

editMenus

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

defaultMenubar property, height property, menubar property, start editing command, stop editing command, vScroll property, About menus and the menu bar, Why does the top of my stack window disappear when I add a menu bar?

Summary

Specifies that the menu bar appears at the top of the stack window.

Syntax

set the editMenus of stack to {true | false}

Example Code

```
set the editMenus of this stack to true
```

Comments

Use the editMenus property to get access to the menu bar for editing on Mac OS systems.

Value:

The editMenus of a stack is true or false.

By default, the editMenus property of newly created stacks is set to false.

Comments:

On Mac OS systems, the menu bar appears at the top of the screen. On Unix and Windows systems, the menu bar appears at the top of the stack window.

If the editMenus property is set to true, the group specified in the stack's menubar property is displayed at the top of the window, and the stack window is resized to accommodate it. If the editMenus property is set to false, the stack window is scrolled and resized so that the menu bar group does not appear in the window. The vScroll of a stack reports the amount the stack has been scrolled down to accommodate a menu bar.

If the stack's menubar property is empty, the editMenus property has no effect.

editScript

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

edit command, errorDialog message, insert script command, scriptParsingError message

Summary

Sent to the current card when the object's script is opened with the edit command.

Syntax

editScript objectID

Example Code

```
on editScript theObject -- save current script before editing
    set the oldScript of theObject to the script of theObject
    pass editScript
end editScript
```

Comments

Handle the editScript message if you want to intercept attempts to edit a script via the message box or a handler.

Parameters:

The objectID is the long ID property of the object whose script is about to be opened.

Comments:

The editScript message is sent when you use the edit command in a handler. However, it is not sent when the development environment is active.

editScripts

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

edit command, password property

Summary

Has no effect and is included in Transcript for compatibility with imported SuperCard projects.

Syntax

set the editScripts to {true | false}

Example Code

Comments

In SuperCard, the editScripts property determines whether scripts can be changed in the script editor. In Revolution, scripts are always editable.

The editScripts property is always set to true. A handler can set it to any value without causing a script error, but the actual value is not changed.

effective

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

owner property, How to find out the actual color of an object, How to restore the default colors of an object, Recipe for automatically updating a Text menu, Text menu > Font > Use Owner's Font, Text menu > Size > Use Owner's Size, Text menu > Color > Use Owner's Color

Summary

Used with inherited properties to specify the object's own setting or the setting inherited from the object's owner, whichever is actually displayed.

Syntax

Example Code

```
put the effective textColor of me into myColor
```

Comments

Use the effective keyword to get the displayed color or font of an object, regardless of whether the object itself has that property set. The effective keyword can also be used to get the filename of a substack or the current rectangle of an object.

Comments:

If one of an object's text, color, or pattern properties is set to empty, the owner's setting is used instead. (If the owner's setting is empty, the setting of that object's owner is used.)

For example, if a card button's backgroundColor property is empty, the backgroundColor of the card is used as the button's backgroundColor. You use the effective keyword to get the color that is actually used. If the button's backgroundColor is not set to empty, the effective keyword gets the setting for the button, since that setting is what the button displays as its background color.

The effective keyword can be used with the following inherited properties:

- backgroundColor property
- backgroundPattern property

- ĩ backgroundPixel property
- ĩ bottomColor property
- ĩ bottomPattern property
- ĩ bottomPixel property
- ĩ focusColor property
- ĩ focusPattern property
- ĩ focusPixel property
- ĩ foregroundColor property
- ĩ foregroundPattern property
- ĩ foregroundPixel property
- ĩ hiliteColor property
- ĩ hilitePattern property
- ĩ hilitePixel property
- ĩ topColor property
- ĩ topPattern property
- ĩ topPixel property
- ĩ shadowColor property
- ĩ shadowPattern property
- ĩ shadowPixel property

- ĩ textFont property
- ĩ textHeight property
- ĩ textSize property
- ĩ textStyle property

- ĩ filename of stack property
- ĩ rectangle property

Note: The effective keyword is implemented internally as a property and appears in the propertyNames. However, it cannot be used as a property in an expression, nor with the set command.

Changes to Transcript:

The ability to use the effective keyword with the rectangle property was introduced in version 1.1.

effectRate

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

idleRate property, unlock screen command, visual effect command

Summary

Specifies the speed of visual effects.

Syntax

set the effectRate to milliseconds

Example Code

```
set the effectRate to 500 -- zippy  
set the effectRate to 5000 -- quite s-s-s-l-l-l-o-o-o-w-w-w...
```

Comments

Use the effectRate property to fine-tune the speed of visual effects.

Value:

The effectRate is an integer between zero and 65535.

By default, the effectRate property is set to 2000 (2 seconds).

Value:

You can specify five speeds with the visual effect command: very fast, fast, normal, slow, or very slow. The effectRate property specifies how long a very slow visual effect takes.

eight
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, eighth keyword, hand constant

Summary

Equivalent to the number 8.

Syntax

Example Code

```
set the bottomMargin of the target to eight
```

Comments

Use the eight constant when it is easier to read than the numeral 8.

eighth

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

eight constant, last keyword, middle keyword, number property

Summary

Designates the eighth member of a set.

Syntax

Example Code

```
send the eighth item of messagesList to the eighth card
```

Comments

Use the eighth keyword in an object reference or chunk expression.

Comments:

The eighth keyword can be used to specify any object whose number property is 8. It can also be used to designate the eighth chunk in a chunk expression.

The word the is optional when using the eighth keyword.

element

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

each keyword, item keyword, repeat control structure

Summary

Used with the repeat control structure to specify that the statements in the repeat loop will be executed once for each element in an array.

Syntax

Example Code

```
repeat for each element thisMonth in monthlyReceivables
```

Comments

Use the element keyword to perform an action on all the elements of an array.

Comments:

The word after the phrase for each element is a variable name. The value of each element is placed in this variable during that iteration of the repeat loop.

The structure

```
repeat for each element myElement in myArray
```

```
...
```

```
end repeat
```

is equivalent to

```
repeat with x = 1 to the number of lines in the keys of myArray
```

```
  put myArray[line x of the keys of myArray] into myElement
```

```
...
```

```
end repeat
```

However, the first form, using the element keyword, is much faster.

else
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

end if keyword, if control structure, then keyword

Summary

Used within an if control structure to contain statements to be performed if the condition is not true.

Syntax

Example Code

```
if it is the date then insertDate else removeDate
```

Comments

Use the else keyword to perform an alternative action within an if control structure.

Comments:

If the condition on the first line of the if control structure is false, the statement or statement list after the else keyword is executed instead of the statements in the main portion of the if control structure. The else section is optional in an if control structure.

Depending on the exact form of the if control structure being used, the else keyword may be on the same line as the rest of the if structure, on the same line as the statement to be executed, or on a line by itself.

emacsKeyBindings

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

keyDown message, lookAndFeel property, Emacs Key Bindings Reference

Summary

Switches key actions between standard actions and those used in the Emacs text editor.

Syntax

set the emacsKeyBindings to {true | false}

Example Code

```
set the emacsKeyBindings to true
```

Comments

Use the emacsKeyBindings property if you're familiar with Emacs and prefer to use its standard keystrokes for text editing.

Value:

The emacsKeyBindings is true or false.

By default, the emacsKeyBindings property is set to false.

Comments:

Emacs is a text editor which is popular among Unix users. It uses keystrokes and key combinations different from the standard Revolution keys. For example:

- Control-V moves the insertion point down a page
- Control-Y pastes the contents of the clipboard
- Control-A moves the insertion point to the beginning of the line
- Control-F moves the insertion point forward one character
- Delete backspaces over the previous character

(For a complete list of supported key bindings, see the Emacs Key Bindings Reference.)

Important! Keyboard equivalents for menu items take precedence over the Emacs key bindings. If one of the key bindings is already in use as a menu item keyboard equivalent, it cannot be used for text editing. (For this reason, some of the Emacs key bindings do not function as expected in the development environment.)

Changes to Transcript:

In versions before 1.1, the `emacsKeyBindings` property affected Command-key combinations, not Control-key combinations, on Mac OS systems.

empty
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, delete variable command, Recipe for removing boldfacing from a field

Summary

Equivalent to nothing, the empty string "".

Syntax

Example Code

```
put empty into field "Results" -- clears the field
```

Comments

Use the empty constant as an easier-to-read substitute for a quoted string with nothing inside the quotes, "".

Comments:

Putting empty into a field or variable clears it.

Putting empty into a nonexistent variable creates the variable with no content.

enable

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

cantSelect property, disable command, disabled property, enable menu command, enabled property, hide command, traversalOn property

Summary

Enables a control so that it responds to user actions.

Syntax

enable object

Example Code

```
enable the selectedObject  
enable field "Info" of stack "Entry"
```

Comments

Use the enable command to allow an object to respond to mouse clicks or keypresses.

Parameters:

The object is any control in an open stack.

Comments:

The enable command is equivalent to setting the object's enabled property to true.

enable menu

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

disable menu command, disabled property, doMenu command, enable command, enabled property, About menus and the menu bar

Summary

Enables a menu or menu item so that it can be chosen by the user.

Syntax

enable [menuItem itemNumber of] menu {menuName | menuNumber}

enable menuItem itemNumber of button

Example Code

```
enable menu 2
enable menuItem 3 of menu "Text" -- menu in menu bar
enable menuItem 4 of button "Edit"
```

Comments

Use the enable menu command to enable a menu or menu item so it can be chosen.

Parameters:

The itemNumber is the number of a menu item, from top to bottom of the menu. The first menu item is numbered 1. (Horizontal lines count.)

The menuName is the name of a menu in the current menu bar.

The menuNumber is the number of a menu, from left to right.

The button is any button.

Comments:

If a menuItem is specified, only that menu item is enabled; otherwise, the entire menu is enabled.

When used to enable an entire menu at once, the enable menu command applies only to menus in the current menu bar. To disable a menu associated with a button, use the enable command.

On Mac OS systems, the Apple menu does not have a number and cannot be enabled or disabled. Menu numbering starts with the menu to the right of the Apple menu.

Since a Revolution menu bar is implemented as a group of buttons (one button per menu, with the menu items as lines in the button's text property), you can indicate a menu by specifying its button. Disabled menu items have an open parenthesis (at the beginning of the line for that menu item. When used to enable a menu item, the enable menu command removes the open parenthesis from the beginning of the specified line.

enabled

property

Synonyms

Objects

control

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

disabled property, enable command, traversalOn property

Summary

Specifies that an object is active and responds to user action.

Syntax

set the enabled of object to {true | false}

Example Code

```
set the enabled of card button "Work Miracles" to true  
set the enabled of menu "Window" to false
```

Comments

Use the enabled property to find out whether a control can respond to mouse clicks or keyboard presses.

Value:

The enabled of a control is true or false.

By default, the enabled property of a newly created control is true.

Comments:

The enabled property of an object is the logical inverse of that object's disabled property. When the enabled is true, the disabled is false, and vice versa.

An enabled control can receive the focus.

Setting a group's enabled property sets the enabled of each control in the group.

You can set the enabled property of a menu by specifying the menu's name or number. An enabled menu can be displayed, and menu items can be individually enabled or disabled using the enable and disable commands.

enabledTracks

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

mediaTypes property, trackCount property, tracks property

Summary

Specifies which tracks in a QuickTime movie are available to be played.

Syntax

set the enabledTracks of player to tracksList

Example Code

```
set the enabledTracks of player "Arctic" to 3 & return & 2
```

Comments

Use the enabledTracks property to control the user's ability to play a QuickTime movie.

Value:

The enabledTracks is a list of tracks, one per line. Each track is a track ID (a positive integer).

Comments:

A movie can contain multiple tracks intended to be played at the same time (for example, an audio and a video track), or tracks that are separate (for example, an alternative audio track). You can list the tracks in a movie using the tracks property.

end

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

end if keyword, end repeat keyword, end switch keyword, end try keyword, EOF constant, function control structure, getProp control structure, on control structure, setProp control structure

Summary

Marks the end of a handler.

Syntax

Example Code

```
end thisFunction
```

Comments

Use the end keyword to end an on, function, getProp, or setProp control structure.

Comments:

The last line of a handler of any kind always begins with the end keyword, followed by the name of the handler.

end if
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

else keyword, end keyword, end repeat keyword, end switch keyword, end try keyword, if control structure, then keyword

Summary

Ends an if control structure.

Syntax

Example Code

```
end if
```

Comments

Use the end if keyword to mark the end of an if control structure.

Comments:

The end if keyword is used only with certain forms of the if control structure. If the entire if control structure is on one line, or if the else clause is on one line, the end if keyword is not used.

The end if keyword occurs on a line by itself.

end repeat
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

end if keyword, end keyword, end switch keyword, end try keyword, exit repeat control structure, next repeat control structure, repeat control structure

Summary

Ends a repeat control structure.

Syntax

Example Code

```
end repeat
```

Comments

Use the end repeat keyword to mark the end of a repeat control structure.

Comments:

The end repeat keyword occurs on a line by itself.

end switch

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

end if keyword, end keyword, end try keyword, switch control structure

Summary

Ends a switch control structure.

Syntax

Example Code

```
end switch
```

Comments

Use the end switch keyword to mark the end of a switch control structure.

Comments:

The end switch keyword occurs on a line by itself.

end try
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

catch keyword, end if keyword, end keyword, end repeat keyword, end switch keyword, finally keyword, try control structure

Summary

Ends a try control structure.

Syntax

Example Code

```
end try
```

Comments

Use the end try keyword to mark the end of a try control structure.

Comments:

The end try keyword occurs on a line by itself.

endArrow

property

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

arrowSize property, points property, startArrow property

Summary

Specifies whether the last vertex in a polygon graphic has an arrowhead.

Syntax

set the endArrow of graphic to {true | false}

Example Code

```
set the endArrow of graphic ID 4 to false
```

Comments

Use the endArrow property to place an arrowhead at the ending point of an irregular polygon or line graphic.

Value:

The endArrow of a graphic is true or false.

By default, the endArrow property of newly created graphics is set to false.

Comments:

For irregular polygons, the end arrow is placed at the point corresponding to the last line of the graphic's points property.

If the style property of the graphic is not polygon or line, the setting of its endArrow property has no effect.

endFrame

property

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

currentFrame property, frameCount property, startFrame property

Summary

The endFrame property is not implemented and is reserved.

Syntax

Example Code

Comments

endTime

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

duration property, playRate property, playSelection property, selectionChanged message, showSelection property, start command, startTime property, stop command

Summary

Specifies the end of the selected portion of a sound or movie.

Syntax

set the endTime of player to endPoint

Example Code

```
set the endTime of player 1 to 1000
```

Comments

Use the endTime property to set or get the selection in a player.

Value:

The endTime of a player is an integer between zero and the player's duration.

By default, the endTime property of newly created players is set to empty.

Comments:

If the playSelection property is true, only the selected portion plays, so you can use the startTime and endTime properties of the player to play whatever portion of the sound or movie you want.

The endTime is the number of the interval where the selection ends. (The number of intervals per second is specified by the player's timeScale property. The total number of intervals is given in the player's duration property.) If there is no selection, the endTime is empty.

endValue

property

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

showValue property, startValue property, thumbPosition property

Summary

Specifies the value corresponding to a scrollbar's end position.

Syntax

set the endValue of scrollbar to number

Example Code

```
set the endValue of last scrollbar to 100
```

Comments

Use the startValue and endValue properties to set the scale of the scrollbar.

Value:

The endValue of a scrollbar is an integer between zero and 65535.

By default, the endValue property of newly created scrollbars is set to 65535.

Comments:

If the style of the scrollbar is scale, the endValue is the value of the thumbPosition when the scrollbar thumb is all the way to the bottom (for a vertical scrollbar) or right (for a horizontal scrollbar).

If the style of the scrollbar is scrollbar or progress, then when the scrollbar thumb is all the way to the end of the scrollbar, the thumbPosition is the scrollbar's endValue minus the thumbSize.

Setting the endValue to a value less than the scrollbar's thumbSize locks the scrollbar thumb to the top (or left) of the scrollbar, preventing scrolling.

english
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

abbreviated keyword, convert command, date function, dateTime keyword, long keyword, short keyword, system keyword, time function, useSystemDate property

Summary

Used with the date and time functions to specify a date or time in the format used in the United States.

Syntax

Example Code

```
put the English date into field "US Formatted Date"  
if the short English date is savedDate then goToList
```

Comments

Use the english keyword to use a date in an invariant format that is guaranteed not to change when the stack is moved to another system, or when users change their system preferences.

Comments:

You can use the english keyword in combination with the long, abbreviated, or short keywords. The english keyword must come before the long, abbreviated, or short keyword.

A long English date looks like this: Thursday, June 22, 2000

An abbreviated English date looks like this: Thu, Jan 22, 2000

A short English date looks like this: 1/22/00

A long English time looks like this: 11:22:47 AM

An abbreviated English time or short English time looks like this: 11:22 AM

Note: The english keyword is implemented internally as a property and appears in the propertyNames. However, it cannot be used as a property in an expression, nor with the set command.

Changes to Transcript:

The ability to use the short, abbreviated, or long keywords in conjunction with the english keyword was introduced in version 1.1. In previous versions, the english keyword was equivalent to the long keyword.

enterInField

message

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

enterKey message, keyDown message, returnInField message

Summary

Sent to the field with the current selection when the user presses the Enter key.

Syntax

enterInField

Example Code

```
on enterInField -- add up the lines in the current field
  put sum(replaceText(target,return,comma)) into field "Sum"
end enterInField
```

Comments

Handle the enterInField message when you want to perform an action when the user presses Enter in a field.

Comments:

The Enter key is usually located on the numeric keypad. It is a different key from the Return key (confusingly labeled "Enter" on some keyboards), which is usually located above the right-hand Shift key.

If the enterInField handler does not pass the message or send it to a further object in the message path, the keypress has no effect. Passing the message allows the keypress to have its normal effect.

If there is no selection or insertion point in any field and the user presses Enter, the enterKey message is sent instead of enterInField.

enterKey

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

enterInField message, returnKey message, tabKey message, How to make the Return or Enter key activate a button

Summary

Sent when the user presses the Enter key, if there is no text selection.

Syntax

enterKey

Example Code

```
on enterKey -- in a stack script, closes the stack on Enter key
  close me
end enterKey
```

Comments

Handle the enterKey message when you want to perform an action (such as going to the next card) when the user presses Enter .

Comments:

The Enter key is usually located on the numeric keypad. It is a different key from the Return key (confusingly labeled "Enter" on some keyboards), which is usually located above the right-hand Shift key.

The message is sent to the active (focused) control, or to the current card if no control is focused. If there is a text selection or insertion point in a field and the user presses Enter, the enterInField message is sent to that field, and the enterKey message is not sent.

environment

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

buildNumber function, licensed function, lookAndFeel property, machine function, platform function, revLicenseType function, screenVendor function, systemVersion function, version function, How to create a standalone application

Summary

Returns the type of running environment the stack is in.

Syntax

the environment

environment()

Example Code

```
the environment
if the environment is "development" then showDevelopOptions
```

Comments

Use the environment function to change a stack's behavior depending on how the stack was opened.

Value:

The environment function returns one of "development", "helper application", or "standalone application".

Comments:

If the environment function returns "development", the stack is running under the development environment. (To find out which edition of the development environment is running, check the revLicenseType function.)

If the environment function returns "helper application", Revolution is running as a helper application, configured by a web browser to display web-based content.

If the environment function returns "standalone application", the stack is running as a standalone application.

If you start up Revolution by Command-double-clicking a stack (Mac OS or OS X) or Control-double-clicking (Unix or Windows), so that the Revolution development environment is not available, Revolution behaves as a standalone application.

EOF

constant

Synonyms

end, endOfFile

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

read from file command, read from process command, stdin keyword, stdout keyword, write to file command

Summary

Equivalent to the EOT character (ASCII 4, Control-D).

Syntax

Example Code

```
read from file monsterFile until EOF
```

Comments

Use the EOF constant to read an entire file, or to read all data from a process.

Comments:

The EOF constant signifies the end of a file.

When used with the read from process or read from file command, the EOF constant causes Revolution to keep reading data until no more data remains to be read.

EPS

object

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

templateEPS keyword, About object types and object references, File menu > Import As Control > EPS File...

Summary

A control that displays Encapsulated PostScript (EPS).

Syntax

Example Code

```
set the postscript of EPS "Preview" to field "Code"
```

Comments

Use the EPS object type to display a screen preview of a PostScript file, or to show PostScript graphics.

Comments:

This object type is supported only on Unix systems with Display PostScript installed.

An EPS object is contained in a card, group, or background. EPS objects cannot contain other objects.

The EPS object has a number of properties and messages associated with it. To see a list of messages that can be sent to an EPS object as a result of user actions or internal Revolution events, open the "Transcript Language Dictionary" page of the main Documentation window, and choose "EPS Messages" from the Show menu at the top. To see a list of all the properties an EPS object can have, choose "EPS Properties" from the Show menu.

eraser
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

choose command, eraser property, tool function, Shortcut to make an image transparent

Summary

Designates the paint tool used to erase portions of the image.

Syntax

Example Code

```
choose eraser tool
```

Comments

Use the eraser keyword to remove paint from an image.

Comments:

When using the Eraser tool, the cursor is the eraser shape (over images) or an arrow (anywhere else). This tool is used to remove paint from an image, making the erased areas transparent.

eraser property

Synonyms

Objects global

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

brush property, choose command, eraser keyword, spray property, tool property

Summary

Specifies the shape used for the Eraser paint tool.

Syntax

set the eraser to {brushID | imageID}

Example Code

```
set the eraser to 12
```

Comments

Use the eraser property to specify which shape is painted by the Eraser tool.

Value:

The eraser is a brush specifier.

A brushID is a built-in brush number between 1 and 35. (These brushes correspond to Revolution's built-in patterns 101 to 135.)

An imageID is the ID of an image to use for erasing. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the eraser is set to 2 (a square eraser).

Comments:

The entire area of the eraser cursor is used as the eraser shape. The shape erased by the eraser is cleared to transparent, regardless of what colors might be in the image used for the eraser shape.

When the Eraser tool is in use, the cursor is the same as the eraser shape. You can use any size image as an eraser, but the cursor may appear distorted on some systems if the image is not 16x16 pixels.

errorDialog

message

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

executionError property, lock error dialogs command, lockErrorDialogs property, scriptParsingError message, How to find out whether a command or function is defined, Recipe for getting the dimensions of a picture file

Summary

Sent to an object when one of its handlers cannot be run due to an execution error.

Syntax

errorDialog errorContents

Example Code

```
on errorDialog myError
end errorDialog
```

Comments

Handle the errorDialog message to prevent the standard error window from appearing, when you want to handle the error in a custom handler.

Parameters:

The errorContents specifies the details of the problem that caused the errorDialog message to be sent, and consists of three lines:

1. Information about the statement that caused the error
2. Information about the handler where the error occurred
3. Information about the object whose script the handler is in

Comments:

If the script contains a compile error, the scriptParsingError message is sent instead of errorDialog.

A list of possible execution errors is contained in the "cErrorsList" property of the first card of the stack "revErrorDisplay". You can view the list with the following statement:

answer the cErrorsList of card 1 of stack "revErrorDisplay"

Note: The exact format of the errorContents may change from release to release.

If the lockErrorDialogs property is true, the errorDialog message is sent to the object that set the lockErrorDialogs., rather than to the object whose script contained the error. (The lockErrorDialogs can be set to true either by setting the property directly, or with the lock error dialogs command.)

Changes to Transcript:

The errorContents parameter was introduced in version 1.1. In previous versions, this information was stored in the executionError property.

errorObject

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

errorDialog message, executionError property, lockErrorDialogs property

Summary

Removed in version 1.1.

Syntax

the errorObject

errorObject()

Example Code

the errorObject

edit script of the errorObject

Comments

The errorObject function returned the name of the object whose script had the most recent execution error.

For the information formerly returned by the errorObject function, use the target function inside an errorDialog handler. The errorDialog message is sent to the object whose script caused the error, so checking the target function will return the name of that object.

escapeKey

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

keyDown message

Summary

Sent when the user presses the Escape key.

Syntax

escapeKey

Example Code

```
on escapeKey -- return to the last card visited
  go recent card
end escapeKey
```

Comments

Handle the escapeKey message if you want to perform some action when the user presses Escape.

Comments:

The message is sent to the active (focused) control, or to the current card if no control is focused.

executionError

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

errorDialog message, errorObject function, lockErrorDialogs property, scriptError property, try control structure

Summary

Removed in version 1.1.

Syntax

get the executionError

set the executionError to empty

Example Code

Comments

The executionError property reported information about the most recent execution error.

For the information formerly reported by the executionError property , see the errorDialog message.

exists

function

Synonyms
existence

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

owner property, there is a operator

Summary

Returns true if the specified object exists, false if it does not.

Syntax

exists(object)

Example Code

```
exists(card field 1)  
exists(button "OK" of card "Preferences")
```

Comments

Use the exists function to make sure an object exists before you try to perform an action on it, or to determine whether to create an object.

Parameters:

The object is any object reference.

Value:

The exists function returns true or false.

Comments:

To find out whether any of a certain type of object exist, use the exists function with the number 1. For example, exists(image 1) returns true if there are any images on the current card, and false otherwise.

You can also specify a chunk of a container, but in this case the exists function always returns true.

Tip: To find out whether a file or folder exists, or whether a process is running, use there there is a operator.

exit

control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

break control structure, exit repeat control structure, exit to top control structure, function control structure, getProp control structure, on control structure, pass control structure, return control structure, setProp control structure, How to stop a running handler

Summary

Stops execution of a handler.

Syntax

exit handler

Example Code

```
if the seconds > timeLimit then exit mouseUp
exit myCustomProperty -- can be used in getProp or setProp handlers
```

Comments

Use the exit control structure to skip the rest of a handler's statements without returning a result.

Form:

The exit statement appears on a line by itself, anywhere inside a handler.

Parameters:

The handler is the name of the handler in which the exit control structure appears.

Comments:

You can use an exit control structure in a message handler, function handler, getProp handler, or setProp handler. Usually, exit is used within an if control structure, so that the handler stops if a condition is true and continues if the condition is false.

If the current handler was called from another handler, the calling handler continues executing. The exit statement only stops the current handler, not the calling handler. (To stop all pending handlers, use the exit to top control structure.)

When a handler executes an exit statement, the message, trigger, function call, or getProp call stops and is not passed to the next object in the message path. To halt the current handler and pass the message, trigger, or call on through the message path, use the pass control structure instead. To halt the current handler and return a result, use the return control structure instead.

Note: The exit control structure is implemented internally as a command and appears in the commandNames.

exit repeat

control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

end repeat keyword, exit control structure, exit to top control structure, next repeat control structure, repeat control structure

Summary

Skips the rest of the current repeat loop and goes to the statement following the end repeat.

Syntax

exit repeat

Example Code

```
if x > 0 then exit repeat
```

Comments

Use the exit repeat control structure to skip the rest of a repeat loop.

Form:

The exit repeat statement appears on a line by itself, anywhere inside a repeat control structure.

Comments:

After an exit repeat statement, none of the remaining statements in the current loop is executed, and any more loops are skipped. The handler resumes executing at the first statement after the end of the repeat loop.

Usually, exit repeat is used within an if control structure, so that the loop stops if a condition is true and continues if the condition is false. This example stops the loop when the user presses the mouse button:

```
repeat with x = 1 to the number of cards
  go card x
  if the mouse is down then exit repeat -- bail out
end repeat
```

exit to top

control structure

Synonyms

exit to MetaCard, exit to SuperCard, exit to HyperCard

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

exit control structure, pass control structure, How to stop a running handler

Summary

Halts the current handler and all pending handlers.

Syntax

exit to top

Example Code

```
if the result is not empty then exit to top -- stop everything
```

Comments

Use the exit to top control structure to stop executing the current handler and suppress pending messages.

Form:

The exit to top statement appears on a line by itself, anywhere inside a handler.

Comments:

Usually, exit to top is used within an if control structure, so that execution stops if a condition is true and continues if the condition is false.

You can use an exit to top statement in a message handler, function handler, getProp handler, or setProp handler.

If the current handler was called from another handler, the calling handler is also halted. Other messages that depend on the same action are also suppressed: for example, if an exit to top control structure is executed in a closeCard handler, the corresponding openCard message is not sent to the destination card.

Important! When a handler executes an exit to top statement, the message, setProp trigger, function call, or getProp call is not passed to the next object in the message path. To halt the current handler and

pass the message, trigger, or call to the next object in the message path, use the pass control structure instead of exit or exit to top.

exitField

message

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeField message, openField message

Summary

Sent to the field with the selection when the selection is being removed from the field, and its contents have not changed.

Syntax

exitField

Example Code

```
on exitField -- remove visual signs that the field is being edited
    set the showBorder of the target to false
end exitField
```

Comments

Handle the exitField message if you want to do something when the user leaves a field that hasn't been changed.

Comments:

The selection is removed from a field (and the field loses focus) when another window is brought to the front, when the user clicks in another field, or when the user tabs out of the field. The field also loses focus when the select command is used to select text in another field. However, the exitField message is not sent when the user clicks another point in the same field.

The exitField message is sent to buttons whose menuMode is "comboBox", since the type-in box in a combo box behaves like a field.

The exitField message is sent only if the field's contents have not changed since the last time it received the openField message. If a field is closing and its contents have changed, the closeField message is sent instead of exitField. This means that if you want to take an action when the selection is removed from a field whether the field has changed or not you must handle both closeField and exitField.

exp function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

^ operator, baseConvert function, exp1 function, exp10 function, exp2 function, ln function

Summary

Returns the natural exponential of a number.

Syntax

the exp of number
`exp(number)`

Example Code

```
exp(1) -- returns e  
exp(theReturnedValue)
```

Comments

Use the exp function to obtain a power of e, e^{number} .

Parameters:

The number is a real number, or an expression that evaluates to a number.

Value:

The exp function returns a positive number.

Comments:

The transcendental number e appears in many mathematical formulas.

exp1 function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

^ operator, baseConvert function, exp function, exp10 function, exp2 function, ln1 function

Summary

Returns the natural exponential of a number, minus 1.

Syntax

the exp1 of number
`exp1(number)`

Example Code

```
exp1(myValue)
```

Comments

Use the exp1 function to obtain a power of e minus 1, $e^{\text{number}} - 1$.

Parameters:

The number is a real number, or an expression that evaluates to a number.

Value:

The exp1 function returns a positive number.

Comments:

The transcendental number e appears in many mathematical formulas.

exp10 function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

^ operator, baseConvert function, exp function, exp2 function, log10 function

Summary

Returns the decimal exponential of a number.

Syntax

the exp10 of number
`exp10(number)`

Example Code

```
exp10(2) -- returns 100  
exp10(-3) -- returns 0.001
```

Comments

Use the exp10 function to obtain a power of 10.

Parameters:

The number is a real number, or an expression that evaluates to a number.

Value:

The exp10 function returns a positive number.

Comments:

The expression `exp10(number)` is equal to 10^{number} .

If number is a positive integer, `exp10(number)` is a 1 followed by number zeros. If number is a negative integer, `exp10(number)` is a decimal point followed by number-1 zeros and a one.

exp2 function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

\wedge operator, baseConvert function, exp function, log2 function

Summary

Returns the binary exponential of a number.

Syntax

the exp2 of number
`exp2(number)`

Example Code

```
exp2(8) -- returns 256  
exp2(the screenDepth) -- returns number of colors available
```

Comments

Use the exp2 function to obtain a power of 2.

Parameters:

The number is a real number, or an expression that evaluates to a number.

Value:

The exp2 function returns a positive number.

Comments:

The expression `exp2(number)` is equal to 2^{number} .

expanded

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

condensed keyword, plain keyword, textStyle property

Summary

Used with the textStyle property to indicate horizontally extended text.

Syntax

Example Code

```
set the textStyle of field "Title Bar" to expanded  
if the textStyle of the mouseControl is "condensed" then beep
```

Comments

Use the expanded keyword to increase the width of text without changing its size.

Comments:

You can extend an object (which extends all the text displayed by the object) or a chunk in a field or button.

To extend text without removing other styles (such as bold or italic), add the expanded keyword to the end of the textStyle. The following example extends the text of a field without affecting its existing textStyle:

```
if the textStyle of field "List" is empty then  
    set the textStyle of field "List" to expanded  
else  
    set the textStyle of field "List" to  
  
    (the textStyle of field "List") & comma & "expanded"  
end if
```

explicitVariables

property

Synonyms

explicitVars

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

local command, localNames function, scriptError property

Summary

Specifies that local variables must be explicitly declared with the local command.

Syntax

set the explicitVariables to {true | false}

Example Code

```
set the explicitVariables to true
```

Comments

Set the explicitVariables property to aid in debugging your code.

Value:

The explicitVariables is true or false.

By default, the explicitVariables is set to false.

Comments:

Setting the explicitVariables property to true can help you debug certain problems with variables.

If the explicitVariables is true, using a local variable without declaring it first causes a compile error. This behavior can be useful in tracking down certain subtle problems such as misspelling a variable name.

If the explicitVariables property is true, using a literal string without enclosing it in quotes causes a compile error.

export

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

export snapshot command, import command, JPEGQuality property, paint keyword, Image Types Reference, How to create a file, How to create a screenshot of a stack, How to import and export styled text, How to store preferences or data for a standalone application, Why can't I export an image to a GIF file?

Summary

Exports the selected image as a PBM, JPEG, GIF, or PNG file.

Syntax

```
export [format] to {file filePath | container} [with mask maskFile]
export image to {file filePath | container} [as format] [with mask maskFile]
```

Example Code

```
export paint to file "Picture" with mask "../circlemask"
export JPEG to URL "binfile:next.jpg"
export image "Parachute" to myVariable as GIF
export image thisImage to file "Thumbnail" as PNG
```

Comments

Use the export command to export an image to a file or container.

Parameters:

The format is one of the following: paint, JPEG, GIF, or PNG.

The filePath specifies the name and location of the file you want to export to. If you specify a name but not a location, the file is created in the defaultFolder.

The container is a reference to a container, usually another image or a URL.

The maskFile specifies the name and location of a file to export as an image mask. You can use a maskFile only when exporting in PBM format (paint).

The image is a reference to an image.

Comments:

If you use the form export format to..., the selected image is exported.

The export paint form exports the image as a PBM, PGM, or PPM file, depending on the image's colors. (Optionally, you can specify a location for the mask file.) The export JPEG form exports as a JPEG file, and the export PNG form exports as a PNG file. If you don't specify a format, the file is exported as PBM, PGM, or PPM.

Tip: If the image has an alpha channel, the export PNG form creates a 32-bit PNG file.

Important! To use the export GIF form, you must purchase a special license key that includes the right to use the LZW compression patent. (Contact support@runrev.com for information.)

Changes to Transcript:

The GIF export option was introduced in version 1.1. In previous versions, it was not possible to export GIFs.

The export image form was introduced in version 1.1. In previous versions, only the selected image could be exported.

The export to container form was introduced in version 1.1. In previous versions, an image could be exported only to a file.

export snapshot

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

export command, import snapshot command, rectangle property, screenRect function, windowID property, How to create a screenshot of a stack, Why can't I export an image to a GIF file?, File menu > Import As Control > Snapshot

Summary

Creates a picture file from a portion of the screen.

Syntax

export snapshot [from rect[angle] rectangle [of window windowID]]

to {file filePath |container} [as format] [with mask maskFile]

Example Code

```
export snapshot to file "Test.ppm" -- displays crosshairs to select area
export snapshot from rect "0,0,200,200" to file "Nav.jpg" as JPEG
export snapshot to pictVariable as GIF
```

Comments

Use the export snapshot command to export a screenshot to a file or container.

Parameters:

The rectangle specifies the edges of the rectangle to be imported, separated by commas, in the same order as the rectangle property (left, top, right, bottom).

Important! If a window is specified, the rectangle is given in relative (window) coordinates; otherwise, it is given in absolute coordinates.

The windowID is the windowID property of an open stack window. (This is not the same as the stack's ID property.)

The filePath specifies the name and location of the file you want to export to. If you specify a name but not a location, the file is created in the defaultFolder.

The container is a reference to a container, usually an image or a URL.

The format is one of the following: paint, JPEG, GIF, or PNG.

The maskFile specifies the name and location of a file to export as an image mask. You can use a maskFile only when exporting in PBM format (paint).

Comments:

If you use the form export format to..., the selected image is exported.

The export snapshot...as paint form exports the image as a PBM, PGM, or PPM file, depending on the screen depth. (Optionally, you can specify a location for the mask file.) The export snapshot...as JPEG form exports as a JPEG file, and the export snapshot...as PNG form exports as a PNG file. If you don't specify a format, the file is exported as PBM, PGM, or PPM.

Tip: If the image has an alpha channel, the export snapshot...as PNG form creates a 32-bit PNG file.

Important! To use the export snapshot...as GIF form, you must purchase a special license key that includes the right to use the LZW compression patent. (Contact support@runrev.com for information.)

If you don't specify a rectangle, Revolution displays a crosshairs cursor. Click at one corner of the rectangle to be imported and drag to the opposite corner to select the area.

If you specify a windowID, the rectangle's coordinates are relative to the top left corner of the window you specify. However, if the window is partly overlapped by another window, whatever is visible on the screen within that rectangle is placed in the snapshot. In other words, you cannot take a snapshot of a part of a window that is hidden by another overlapping window.

extendKey

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

charToNum function, keyDown message, numToChar function, optionKey function, rawKeyDown message, shiftKey function

Summary

Specifies the key used to create special (extended ASCII) characters on Unix systems.

Syntax

set the extendKey to keyNumber

Example Code

```
set the extendKey to 6 -- disables entry of high-bit chars
```

Comments

Use the extendKey property to specify how to create special characters, or to prevent users from typing special characters.

Value:

The extendKey is a positive integer.

By default, the extendKey is set to 5, so the modifier key is the mod5 key specified by the "xmodmap" program on your computer.

Comments:

The extendKey specifies the modifier key used to toggle the high bit of characters on and off. Use the "xmodmap" program to find out which key is used for mod5 (or the modifier key specified for the extendKey).

To enter a high-bit character, press the modifier key, then type the key that corresponds to that character, then press the modifier key again to return to entering regular ASCII characters.

If the extendKey is greater than 5, entry of high-bit characters is disabled.

The setting of this property has no effect on Mac OS and Windows systems.

extents

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

max function, min function, round function, sum function

Summary

Returns a list of the smallest and largest keys for each dimension of an array.

Syntax

the extents of array

extents(array)

Example Code

```
the extents of myArray
```

```
put item 1 of line 2 of the extents of theFigures into columnsMinimum
```

Comments

Use the extents function to find the minimum and maximum row and column numbers of an array whose keys are integers.

Parameters:

The array is any array of any dimension whose keys are numbers.

Value:

The extents function returns one line for each dimension of the array.

Each line contains two numbers separated by commas. The first item is the lowest key number in that dimension, and the second item is the highest key number.

Comments:

If any of the keys of the array is non-numeric, the extents function returns empty.

externalCommands

property

Synonyms

Objects

stack

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

destroyWindow property, externalFunctions property, externalPackages property, externals property

Summary

Lists the available external commands for a stack.

Syntax

get the externalCommands of stack

Example Code

```
answer the externalCommands of stack "Basics"  
if myExternal is among the lines of the externalCommands  
    of this stack then do myExternal
```

Comments

Use the externalCommands property to find out whether a particular external command is available for use in handlers, or to list all the available external commands.

Value:

The externalCommands of a stack reports a list of available external commands, one per line.

This property is read-only and cannot be set.

Comments:

You use an external command in a handler the same way you use one of Transcript's built-in commands.

Changes to Transcript:

Support for using the externalCommands property on OS X systems was added in version 2.0.

externalFunctions

property

Synonyms

Objects

stack

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

destroyWindow property, externalCommands property, externalPackages property, externals property

Summary

Lists the available external functions for a stack.

Syntax

get the externalFunctions of stack

Example Code

```
put the externalFunctions of this stack after functionsList
if "measures" is among the lines of the externalFunctions

  of this stack then get measures(field 1)
```

Comments

Use the externalFunctions property to find out whether a particular external function is available for use in handlers, or to list all the available external functions.

Value:

The externalFunctions of a stack reports a list of available external functions, one per line.

This property is read-only and cannot be set.

Comments:

You use an external function in a handler the same way you use one of Transcript's built-in functions.

Changes to Transcript:

Support for using the externalFunctions property on OS X systems was added in version 2.0.

externalPackages

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

destroyWindow property, externalCommands property, externalFunctions property, externals property

Summary

Lists the available external packages for a stack.

Syntax

get the externalPackages of stack

Example Code

```
put the externalPackages of stack "Main" into field "Package List"
```

Comments

Use the externalPackages property to find out whether a particular external package is available for use in handlers, or to list all the available external packages.

Value:

The externalFunctions of a stack reports a list of available external packages, one per line.

This property is read-only and cannot be set.

externals

property

Synonyms

Objects

stack

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

destroyWindow property, externalCommands property, externalFunctions property, externalPackages property

Summary

Specifies a list of files containing external code that are to be loaded into memory when the stack opens.

Syntax

set the externals of stack to filePathsList

Example Code

```
set the externals of stack "Apps" to field "Executables"
```

Comments

Use the externals property to use the externals in the specified files.

Value:

The externals of a stack reports a list of file paths, one per line.

Comments:

Each line of the externals specifies the name and location of a file containing external commands and external functions. If you specify a name but not a location, the file is assumed to be in the defaultFolder.

The files are executed when the stack is opened, making the externals in them available to handlers in the stack.

Important! When you install a new external by setting a stack's externals property, Revolution cannot use it until you either quit Revolution and then reopen the stack, or close the stack (after setting its destroyWindow property to true) and then reopen it.

Changes to Transcript:

Support for using the externals property on OS X systems was added in version 2.0.

false

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command, not operator, true constant

Summary

Equivalent to the string "false".

Syntax

Example Code

```
set the showLines of field "Tryout" to false  
if the filled is false then set the filled to true
```

Comments

Use the false constant to indicate that a property is turned off or that a logical expression evaluates to false.

family property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hilitedButton property, radioButton property, selectedButton function

Summary

Coordinates a set of radio buttons so that only one button of a family can be highlighted.

Syntax

set the family of button to number

Example Code

```
set the family of button "Choice #1" to 3
```

Comments

Use the family property to set up a cluster of related buttons.

Value:

The family of a button is an integer.

By default, the family property of newly created buttons is set to zero (no family).

Comments:

Setting the family of a button to a number makes the button a member of that family. All the buttons in a card or group whose family property is the same behave as a radio-button cluster: only one button in the family can be highlighted at a time, and highlighting another member of the family unhighlights all the others.

If a button's family property is zero, it is not a member of any family.

If the buttons are part of a group, the group's `backgroundBehavior` property must be set to true.

This property exists to aid compatibility with imported HyperCard stacks. Unlike the property in HyperTalk, it has no effect on buttons whose style property is not "radioButton".

Tip: You can also create a radio-button cluster by putting the buttons in a group and setting the group's `radioBehavior` property to `true`.

field

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

button keyword, choose command, graphic keyword, image keyword, player keyword, scrollbar keyword, style property, templateField keyword, tool function

Summary

Designates the Field tool, which is used to create new fields.

Syntax

Example Code

```
choose field tool
```

Comments

Use the field keyword to create fields.

Comments:

When using the Field tool, the cursor is a crosshairs. This tool is used for creating fields by clicking at one corner of the field and dragging to the opposite corner.

You can set the style (and other properties) of the templateField to the field type you want, then draw the field with the Field tool. The field icons on the Tools palette work this way: each one sets the style of the templateField, then chooses the Field tool so you can create the field.

field object

Synonyms fld

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

templateField keyword, About object types and object references, Object menu > New Control > Field, Object menu > New Control > Scrolling Field, Object menu > New Control > List Field, Object menu > New Control > Scrolling List Field, Object menu > New Control > Label Field

Summary

A control that contains text.

Syntax

Example Code

```
set the backgroundColor of field "Warnings" to "red"  
put it into line 3 of field "Help"
```

Comments

Use the field object type to hold text or to provide a place for the user to enter text.

Comments:

Fields can contain styled text (with different fonts, sizes, styles, and colors for different parts of the text they contain). A field can be displayed with or without horizontal and vertical scrollbars, and can be unlocked (allowing the user to enter text) or locked.

A field is contained in a card, group, or background. Fields cannot contain other objects.

The field object has a number of properties and messages associated with it. To see a list of messages that can be sent to a field as a result of user actions or internal Revolution events, open the "Transcript Language Dictionary" page of the main Documentation window, and choose "Field Messages" from the Show menu at the top. To see a list of all the properties a field can have, choose "Field Properties" from the Show menu.

fifth

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

five constant, last keyword, middle keyword, number property

Summary

Designates the fifth member of a set.

Syntax

Example Code

```
copy the fifth button of the second group  
select fifth line of field "Address Info"
```

Comments

Use the fifth keyword in an object reference or chunk expression.

Comments:

The fifth keyword can be used to specify any object whose number property is 5. It can also be used to designate the fifth chunk in a chunk expression.

The word "the" is optional when using the fifth keyword.

file

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

binfile keyword, close file command, http keyword, open file command, read from file command, resfile keyword, revCopyFile command, URL keyword, write to file command, About filename specifications and file paths, About using URLs, uploading, and downloading, How to associate files with a Mac OS standalone application, How to associate files with an OS X standalone application, How to associate files with a Windows standalone application, How to create a file, How to delete a file, How to rename a file, Why is there a problem with line endings?

Summary

Used as a URL type with such commands as put and get to designate a local file.

Syntax

Example Code

```
put the htmlText of field 1 into URL "file:/Drive/Folder/File"  
answer URL "file:myPrefs.txt"  
put URL "file:C:/abc.def" into savedInfo  
put "Test" into word 3 of URL "file:/Hard Drive/Applications/Test"
```

Comments

Use the file keyword to work with text files.

Comments:

The file scheme indicates a text file which is located on the user's system. The file is specified by either a full path starting with "/", or a relative path starting from the defaultFolder.

A URL container can be used anywhere another container type is used.

Different operating systems use different characters to mark the end of a line. Mac OS and OS X use a return character (ASCII 13), Unix systems use a linefeed character (ASCII 10), and Windows systems use a return followed by a linefeed. When you use a file URL as a container., Revolution automatically uses the current system's standard end-of-line marker and Revolution's linefeed character.

Tip: To put data into, or get data from, a binary file, use the `binfile` keyword instead.

For technical information about URLs and the file URL scheme, see RFC 1630 at [<http://www.ietf.org/rfc/rfc1630.txt>](http://www.ietf.org/rfc/rfc1630.txt).

filename

property

Synonyms

Objects

image, player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bufferHiddenImages property, constantMask property, filename of stack property, import command, name property, play command, tool property, URL keyword, About filename specifications and file paths, Image Types Reference, How to display a picture from a web server, How to display the contents of a text file, How to play a streaming QuickTime file, File menu > New Referenced Control

Summary

Specifies the file associated with a player or image.

Syntax

set the filename of {image | player} to {filePath | URL}

Example Code

```
set the filename of image "Animation" to "/HD/Projects/blipvert.gif"  
set the filename of last player to "http://www.example.com/myMovie.mov"  
set the filename of image ID 10 to "binfile:Picture Example"
```

Comments

Use the filename property to reference a separate file that holds the data for an image or player that's a referenced control.

Value:

The filename of an image or player specifies either a file path or a URL. A file path specifies the name and location of the file. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder. A URL specifies the name and location of a file on your system or anywhere on the Internet.

By default, the filename property of newly created images and players is set to empty.

Comments:

The contents of an image can reside either in the stack or in a separate file. If an image's filename is empty, the image uses its own data, which resides in the stack, instead of loading a separate file. (You

can import an image with the import command, or paint an image with the paint tools.) Setting an image's filename property to the name and location of a file deletes the previous contents of the image.

Using an image from a separate file reduces memory use, since the image does not need to be loaded into memory unless it is in use. But it increases the time it takes to go to a card containing the image, since the file must be loaded from the disk it's on or from the Internet.

The contents of a player must reside in a separate file that is specified in the player's filename property. If a player's filename is empty, the player does not display any data. The file specified by the filePath or URL must be in a format supported by QuickTime.

Important! Setting a player's filename does not automatically update its currentTime property. If you play a movie file, then change the player's filename in order to play another, you must reset the currentTime to zero in order to start from the beginning of the second movie:

```
set the filename of player "My Player" to "SecondMovie.mov"  
set the currentTime of player "My Player" to zero
```

If you specify a URL for a streaming QuickTime movie file, Revolution displays the movie sequentially as it downloads. Otherwise, Revolution must download the entire file before you can play it. To pre-fetch a file from the Internet in order to speed up access to it, use the load command before visiting the card that holds the player or image that references the file's URL.

Important! Don't use the URL keyword when specifying a URL. The filename of an image or player is a file location, not the image or movie data itself. If you use the URL keyword, the filename property is set to the contents of the URL, not the URL itself, and this is usually not what's wanted.

You can use the "New Referenced Control" submenu in the File menu to create a new image or player which references the contents of a file you specify. You can also set the filename to an absolute path or relative path by changing the Source field in the image's or player's property inspector.

Changes to Transcript:

The URL option was introduced in version 1.1. In previous versions, the filename of an image or player always specified a file path on the local system, and files on the Internet could not be used.

filename of stack

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

effective keyword, mainStack property, name property, rename command, revert command, save command, saveStackRequest message, stackFiles property, stackFileType property, substacks property, tempName function, About main stacks, substacks, and the organization of a stack file, About filename specifications and file paths, How to find out the location of the current stack's file

Summary

Reports the file path of the file a stack is stored in.

Syntax

set the filename of stack to filePath

Example Code

```
put the filename of this stack into nameToSave
if the effective filename of this stack is savedName then beep
set the filename of this stack to "Test.rev"
set the filename of this stack to "/Volumes/Lizards/Godzilla.rev"
```

Comments

Use the filename of stack property to specify where a stack file is stored on the user's system.

Value:

The filename of stack of a stack specifies the name and location of the stack file. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

By default, the filename of stack property of newly created stacks is set to empty.

Comments:

If the stack is a main stack, its filename of stack property reports the file path of the file the stack is stored in. If the stack is the main stack of a standalone application, the filename of stack property reports the name and location of the application.

If the stack is a substack, its filename of stack property is empty. To find out what file a substack is stored in, use the form the effective filename of stack. You cannot set the filename of stack property of a substack.

If the stack has not yet been saved, its filename of stack property is empty.

Cross-platform note: On OS X systems, standalone applications are stored as application bundles. A bundle behaves like a file but is actually a folder, and the main stack of a standalone application is inside this folder. The filename of stack property reports the location of the application inside the bundle, not the bundle's location. For example, if the bundle's file path is `"/Volumes/Disk/MyApp.app/"`, the filename of the application's main stack might be `"/Volumes/Disk/MyApp.app/Contents/MacOS/MyApp"`.

files

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

answer file command, convert command, fileType property, folders function, long keyword, open file command, rename command, revCopyFile command, seconds function, there is a operator, umask property, About filename specifications and file paths, How to change the current folder used for file operations, How to determine the size of a file, How to determine whether a file exists

Summary

Returns a list of files in the defaultFolder.

Syntax

the [detailed | long] files
files()

Example Code

```
put the files into field "Current Files"  
repeat with x = 1 to the number of lines of the detailed files  
put the files & the folders into diskContents[the defaultFolder]
```

Comments

Use the files function to obtain a list of files to display for the user, or to perform an action on each file in the current folder.

Value:

The files function returns a list of file names, one per line.

The detailed files form returns a list of files, one file per line. Each line contains the following attributes, separated by commas:

- ï The file's name, URL-encoded
- ï The file's size in bytes (on Mac OS and OS X systems, the size of the file's data fork)
- ï The resource fork size in bytes (Mac OS and OS X systems only)
- ï The file's creation date in seconds (Mac OS, OS X, and Windows systems only)
- ï The file's modification date in seconds
- ï The file's last-accessed date in seconds (Unix and Windows systems only)

- ï The file's last-backup date in seconds (Mac OS and OS X systems only)
- ï The file's owner (Unix systems only)
- ï The file's group owner (Unix systems only)
- ï The file's access permissions
- ï The file's creator and file type (Mac OS and OS X only)

Any attribute that is not supported on the current system is reported as "0" (zero).

Comments:

Folders in the defaultFolder are not included. To get a list of folders, use the folders function.

The names of aliases (on Mac OS and OS X systems), symbolic links (on Unix systems), and shortcuts (on Windows systems) are included in the value returned by the files if they refer to a file. If they refer to a folder, they are not included.

The forms the detailed files and the long files are synonyms.

When listed in the detailed files form, each file's name is URL-encoded. To obtain the name in plain text, use the URLDecode function. If the detailed modifier is not used, the filename is not encoded.

The access permissions returned in the detailed files form consist of three octal digits. The form is the same as that used for the umask command.

The creator and file type returned in the detailed files form is an eight-character string. The first four characters are the creator signature, and the last four are the file type.

Changes to Transcript:

The detailed files form was introduced in version 1.1. In previous versions, the files function provided only a list of file names.

fileType

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

binfile keyword, file keyword, filename property, files function, open file command, stackFileType property, How to assign a custom creator signature to a standalone, How to assign creator and type signatures to a new file, How to change the file type of an existing file

Summary

Specifies the creator and file type for any non-stack files a handler creates on a Mac OS or OS X system.

Syntax

set the fileType to creator & type

Example Code

```
set the fileType to "ttxTEXT" -- text file owned by SimpleText app
```

Comments

Use the fileType property to ensure that files that a standalone application creates are recognized by the operating system as belonging to the standalone.

Value:

The fileType is an eight-character string. The first four characters are the creator signature, and the last four are the file type.

Important! The file type and creator signature are case-sensitive.

Comments:

When a file is saved on a Mac OS or OS X system, a 4-character file type and 4-character creator signature are saved with it. The creator signature specifies which application owns the file. The application determines the file format from the file type; the type is also used to determine which applications (other than the owner) can work with the file.

The fileType property is used to set the file type and creator of files created by the open file command and of files created putting data into a file, binfile, or resfile URL that doesn't yet exist. (To specify the

file type and creator for stack files your application creates with the save command, use the `stackFileType` property instead.)

This property has no effect on Unix and Windows systems.

Important! To avoid conflicts with other applications, register any new creator signatures with Apple Computer if you plan to distribute a stack or standalone application that uses this property. Apple maintains a registry of creator signatures on its web site at <http://developer.apple.com/dev/cftype/>.

filled

property

Synonyms
showFill

Objects
graphic, global

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
brushColor property, brushPattern property, centered property, noOp keyword, opaque property

Summary
Specifies whether graphics and shapes drawn with the paint tools are filled or hollow.

Syntax
set the filled [of graphic] to {true | false}

Example Code

```
set the filled to true -- affects paint tools
set the filled of graphic "Square" to false
```

Comments
Use the filled property to change the appearance of graphics

Value:
The filled of a graphic is true or false.

By default, the filled of newly created graphics is set to false.

Comments:
If the filled property is set to true, the interior of shapes is filled with the brushColor (or brushPattern, if one is set). If the graphic's style property is "line", the filled property has no effect.

The global setting of the filled property controls the appearance of shapes drawn with the paint tools. Once a paint shape is drawn, its appearance cannot be changed by changing the global filled property.

Tip: Clicking the interior of a graphic whose filled is set to false does not send mouse messages to the graphic unless the graphic is already selected. To create a graphic whose interior is transparent, but clickable, set the graphic's filled property to true and its ink property to noOp.

filter

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

caseSensitive property, matchChunk function, matchText function, replace command, sort command, Regular Expressions Syntax Reference

Summary

Filters each line in a container, removing the lines that do or don't match a wildcard expression.

Syntax

filter container {with | without} wildcardExpression

Example Code

```
filter myVariable with "A?2"  
filter me without "[a-zA-Z]*"   
filter field 22 with "[0-9]*"   
filter field "Sorted Lines" with "[" & mySelection & "]"
```

Comments

Use the filter command to pick specific lines in a container.

Parameters:

The container is any expression that evaluates to a container.

The wildcardExpression is a pattern used to match certain lines.

Comments:

The filter...with form places the lines that contain a match for the specified wildcard expression in the container, replacing the previous contents.

The filter...without form places the lines that do not contain a match for the specified wildcard expression in the container, replacing the previous contents.

Wildcard expressions are similar to regular expressions. You can use the following characters in a wildcard expression:

*

Matches zero or more of any character. The wildcard expression A*C matches "AC", "ABC", or "ADZXC".

?

Matches exactly one character. The wildcard expression A?C matches "ABC", but not "AC" or "ADZXC".

[chars]

Matches any one of the characters inside the brackets. The wildcard expression A[BC]D matches "ABD" or "ACD", but not "AD" or "ABCD".

[char-char]

Matches any character whose ASCII value is between the first character and the second character.

Changes to Transcript:

The filter...without form was added in version 2.1.1. In previous versions, the filter command could be used only to retrieve lines that matched a wildcard expression.

finally
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

else keyword, try control structure

Summary

Used within a try control structure to contain statements that are executed whether or not there was an error.

Syntax

Example Code

`finally`

Comments

Use the finally keyword to designate one or more statements that are to be executed within a try control structure.

Comments:

The finally keyword appears on a line by itself, and is followed by a list of statements. The statements in the finally section are executed whether or not there was an error while executing the try control structure.

The finally section is always executed even if the try control structure contains an exit or pass statement, so it can be used for final cleanup actions such as deleting variables.

The finally section is optional in a try control structure.

find

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

caseSensitive property, dontSearch property, foundChunk function, foundField function, foundLine function, foundLoc function, foundText function, mark command, matchText function, normal keyword, offset function, replace command, How to remove the box around found text, How to search a container, How to search a stack, Recipe for a Find field, Edit menu > Find and Replace...

Summary

Searches the fields of the current stack.

Syntax

find [form] textToFind [in field]

find empty

Example Code

```
find "heart"  
find string "beat must go on" in field "Quotes"
```

Comments

Use the find command to search for text.

Parameters:

The form is one of the following words:

ī normal

ī characters or character (or chars or char)

ī words or word

ī string

ī whole

If no form is specified, the find normal form is used.

The textToFind is any expression that evaluates to a string.

The field is any expression that evaluates to a field reference. If the field is not specified, the find command searches all the fields in the current stack (except fields whose dontSearch property is set to true).

Comments:

The find command starts searching after the previously-found text (if there was a previous find command) or at the beginning of the first field on the current card (if not).

When the search is successful, the card containing the text appears and a box is drawn around the found text. If the text is not found in the stack, the result function returns "Not found".

The six forms of the find command search in different ways. The find normal, find chars, and find words forms search for each word in the textToFind, but the words don't have to be together to be found; they only have to all appear on the same card. The find words and find whole forms look for complete words only. The find string and find whole forms search for the textToFind as a unit.

The find normal form looks for each word in the textToFind at the beginning of a word. For example, find "ring bell" finds "ringing" and "belltower", but not "bring" or "Campbell". All the words you specify must be in fields on the card, but they don't need to be in the same order, or even in the same field.

The find chars form looks for each word in the textToFind, in any part of a word. Unlike the find normal form, the find chars form doesn't require that the words in the textToFind be found at the start of a word on the card. For example, find chars "ring bell" finds "bring", "ringing", "belltower", and "Campbell". As with the find normal form, all the words must be somewhere on the card, but they don't need to be in the same order, or in the same field.

The find words form looks for each word in the textToFind. All the words must be complete words, not parts of words. For example, find words "ring bell" finds "ring" and "bell", but not "ringing", "bring", "belltower", or "Campbell". As with the find normal and find chars forms, all the words must be somewhere on the card, but they don't need to be in the same order, or in the same field.

The find string form looks for the entire textToFind as a unit. Unlike the find normal, find chars, and find words forms, the find string form requires that the textToFind be found exactly: the words must be in the same order and in the same field, and not separated by other words. For example, find string "ring bell" finds "ring bell" and "bring belltower", but not "ring the bell" (extra word between "ring" and "bell"), "Ringbell Street" (no space between "ring" and "bell"), or "bell ringer" (words are in the wrong order).

The find whole form looks for the entire textToFind as a unit. Like the find words form (and unlike the find string form), the find whole form requires that each word in the textToFind be found as a whole word, not part of a word. For example, find whole "ring bell" finds "ring bell", but not "bring belltower" (the "ring" and "bell" are parts of words, not whole words), "ring the bell" (extra word between "ring" and "bell"), "Ringbell Street" (no space between "ring" and "bell"), or "bell ringer" (words are in the wrong order).

Note: Because the find normal, find words, and find whole forms search for words or portions of words, they cannot find a string containing a space. The find string form can find a string containing a space, but cannot find a string that contains a return character.

The find empty form of the find command removes the box from the last word found and resets the find command, so that the next search starts from the beginning of the current card, rather than the location of the next find. Going to another card also resets the find command.

The setting of the caseSensitive property determines whether the search considers uppercase and lowercase characters to be equivalent.

Note: The search does not consider characters that differ by a diacritical mark to be equivalent. For example, find "mÈre" will not find the word "mere".

Usually, the offset and matchText functions are faster than the find command. But unlike these functions, the find command can search all the fields of a stack at once, instead of one container at a time.

Tip: To perform a batch search, set the mark property of all cards where a match is found by using the mark cards by finding form of the mark command.

first
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

any keyword, last keyword, middle keyword, number property, one constant

Summary

Designates the first member of a set.

Syntax

Example Code

```
go to the first card of stack "Games Central"  
put "a" into first character of theLetters
```

Comments

Use the first keyword in an object reference or chunk expression.

Comments:

The first keyword can be used to specify any object whose number property is 1. It can also be used to designate the first chunk in a chunk expression.

The word the is optional when using the first keyword.

firstIndent

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

leftMargin property, tabStops property, textAlign property

Summary

Determines the indentation of the first line of each paragraph in a field.

Syntax

set the firstIndent of field to pixels

Example Code

```
set the firstIndent to 10
```

Comments

Use the firstIndent property to create indented paragraphs.

Value:

The firstIndent of a field is an integer.

By default, the firstIndent property of newly created fields is set to zero.

Comments:

The first line of each paragraph is left-indented the specified number of pixels. If the firstIndent is zero, the field's paragraphs are not indented. If the firstIndent is negative, the first line of each paragraph is outdented the specified number of pixels, creating a hanging indent. (It may be necessary to increase the field's leftMargin property to accommodate the hanging indent.)

If the field's textAlign property is "center" or "right", the first line of each paragraph starts at least firstIndent pixels from the left margin of the field, but the effect of indentation may not be readily visible unless the field's textAlign is set to "left".

five
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, fifth keyword

Summary

Equivalent to the number 5.

Syntax

Example Code

```
multiply field "Raw Score" by five
```

Comments

Use the five constant when it is easier to read than the numeral 5.

fixedLineHeight

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

formattedHeight property, height property, showLines property, textHeight property

Summary

Specifies whether the lines in a field are all the same height, or vary in height to fit the text on that line.

Syntax

set the fixedLineHeight of field to {true | false}

Example Code

```
set the fixedLineHeight of the selectedField to false
```

Comments

Use the fixedLineHeight property to control the appearance of fields with multiple fonts, sizes, or styles.

Value:

The fixedLineHeight of a field is true or false.

By default, the fixedLineHeight property of newly created fields is set to true.

Comments:

If a field's fixedLineHeight property is set to true, the height allotted for each line in the field is equal to the field's textHeight. This means that if some of the text in the field is taller than the field's textHeight, it may be truncated.

If the fixedLineHeight is false, the height of each line depends on the size of the text in that line.

flip

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

choose command, crop command, imageData property, rotate command, select keyword, Object menu > Flip

Summary

Changes the orientation of an image or the currently selected part of an image.

Syntax

flip [image] {horizontal | vertical}

Example Code

```
flip vertical  
flip image "Logo" horizontal
```

Comments

Use the flip command to turn an image over.

Parameters:

The image is any image reference.

Comments:

The flip vertical form flips the image vertically, swapping its top and bottom. The flip horizontal form flips the image side to side, swapping its right and left.

If you don't specify an image, the portion currently selected with the Select paint tool is flipped.

The flipping operation is its own inverse: flipping an image twice in the same direction restores it to its original appearance.

Important! Flipping a referenced image changes its orientation only temporarily. The next time the referenced image is displayed, its original orientation returns.

flushEvents

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

appleEvent message, cancel command, keyDown message, keyUp message, lockMessages property, mouseDown message, mouseUp message, resumeStack message, suspendStack message, How to throw away unwanted mouse clicks

Summary

Clears pending events from the event queue so they will not trigger handlers.

Syntax

flushEvents(eventType)

Example Code

```
put flushEvents("activate") into trashVar  
get flushEvents("all")
```

Comments

Use the flushEvents function to prevent unwanted messages from being sent during a handler's execution.

Parameters:

The eventType is one of the following:

- all: ignore all waiting events
- mouseDown: ignore mouse presses
- mouseUp: ignore mouse releases
- keyDown: ignore keypresses
- keyUp: ignore key releases
- autoKey: ignore key repeats
- disk: ignore disk-related events
- activate: ignore windows being brought to the front
- highLevel: ignore Apple Events (on Mac OS and OS X systems)
- system: ignore operating system events

Value:

The flushEvents function always returns empty.

Comments:

Typically, you use the flushEvents function in a handler to dump user actions that have occurred during the handler. For example, if a button has a mouseUp handler that takes a few seconds to run, the user might click again during that time. To prevent those extra clicks from causing the handler to run again, use the flushEvents function:

```
on mouseUp
-- ...lengthy handler goes here...
-- get rid of clicks since the handler started:
  put flushEvents("mouseUp") into temp
end mouseUp
```

To clear multiple event types, call the flushEvents function once for each event type you want to clear.

Although some of the eventTypes have the same names as built-in Transcript messages, there is a distinction. For example, the mouseDown event type is the operating system's response to the user clicking the mouse button. When the operating system sends this event to the application, Revolution sends a mouseDown message to the target object. The expression flushEvents(mouseDown) prevents the application from responding to any mouseDown events it has received from the operating system, but has not yet processed.

focus

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

enterKey message, focusColor property, focusedObject function, focusIn message, focusOut message, focusPattern property, keyDown message, keyUp message, lockText property, lookAndFeel property, returnKey message, select command, showFocusBorder property, tabKey message

Summary

Places the insertion point in a field or makes a control active.

Syntax

focus [on] object

Example Code

```
focus on field "Label"  
focus on the mouseControl  
focus on graphic "Move Me"
```

Comments

Use the focus command to make a control active—that is, to make it receive any keystrokes the user types.

Parameters:

The object is any object on the current card.

Comments:

If the object's traversalOn property is false, it cannot receive the focus, and the focus command causes an error.

If the lookAndFeel is set to "Motif", the focused control is outlined, and the control receives any keystrokes (and the messages associated with them).

If the lookAndFeel is set to "Appearance Manager" or "Macintosh", an outline is drawn around fields, images, and EPS objects whose showFocusBorder property is set to true. Otherwise, the appearance of the focused control does not change, but it receives keystroke messages.

If the lookAndFeel is set to "Windows 95", a dotted outline is drawn within buttons when they receive the focus.

If the object is an unlocked field, the insertion point is placed after the text in the field.

focusColor

property

Synonyms
eighthColor

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
backgroundColor property, borderColor property, bottomColor property, colorNames function, colors property, effective keyword, focus command, focusedObject function, focusPattern property, foregroundColor property, hiliteColor property, owner property, shadowColor property, showFocusBorder property, topColor property, About colors and color references, Color Names Reference, Why don't buttons respect my color settings?, Recipe for translating a color name to an RGB numeric triplet

Summary
Specifies the color of the outline around the active control or the field with the insertion point.

Syntax
set the focusColor of object to {empty | colorName | RGBColor}

Example Code
set the focusColor of me to "black"
set the focusColor of button "Help" to 128,128,255

Comments
Use the focusColor property to specify the outline color of an object when it's active.

Value:
The focusColor of an object is any valid color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the focusColor for all objects is empty.

Comments:

Setting the focusColor of an object to empty allows the focusColor of the object's owner to show through. Use the effective keyword to find out what color is used for the object, even if its own focusColor is empty.

If the focusColor is not set for any object in the object hierarchy, the system setting is used.

The setting of the focusColor property has different effects, depending on the object type:

- ī The focusColor of a stack, card, or group determines the focusColor of each object in the stack, card, or group that does not have its own focusColor.
- ī The focusColor of a button is used to outline the button when it is focused. If the button's style is menu, or if the button's traversalOn property is false, the focusColor has no effect.
- ī The focusColor of a field or scrollbar is used to outline the object when it is focused. If the object's traversalOn property is false, the focusColor has no effect.
- ī The focusColor of a graphic, player, audioClip, videoClip, or EPS object has no effect.
- ī The focusColor of an image is the eighth color in the image's color palette.

If an object's focusPattern is set, the pattern is shown instead of the color specified by the focusColor.

The focusColor property has no effect on controls whose showFocusBorder property is set to false.

If the lookAndFeel is set to "Macintosh" or "Appearance Manager", the focusColor affects only fields.

focusedObject

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

enterKey message, focus command, focusColor property, focusIn message, focusOut message, focusPattern property, keyDown message, keyUp message, lockText property, lookAndFeel property, returnKey message, select command, showFocusBorder property, tabKey message

Summary

Returns the currently focused control.

Syntax

the focusedObject

focusedObject()

Example Code

```
the focusedObject  
put the focusedObject into objectToReturnTo
```

Comments

Use the focusedObject function to determine which object receives any keystrokes that are typed by the user or created by the type command.

Value:

The focusedObject function returns the long ID property of the object.

Comments:

The focusedObject can be any control on the current card. If no object has the focus, the focusedObject is the current card.

focusIn

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

focus command, focusOut message, openField message, resumeStack message, showFocusBorder property, traversalOn property

Summary

Sent to a control when it becomes active (focused).

Syntax

focusIn

Example Code

```
on focusIn -- boldface the text of the control
  set the textStyle of the target to "bold"
end focusIn
```

Comments

Handle the focusIn message if you want to perform preparation or do other tasks when a control receives the keyboard focus.

Comments:

If the control is an unlocked field or a button whose menuMode is "comboBox", the openField message is sent to it instead of the focusIn message.

A locked field receives the focusIn message when the user tabs to it or otherwise makes it active (focused), or when text in it is selected by a handler.

focusOut

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeField message, exitField message, focus command, focusIn message, suspendStack message, traversalOn property

Summary

Sent to a button or field when it becomes inactive (loses focus).

Syntax

focusOut

Example Code

```
on focusOut
  -- if you set the textStyle on focusIn, use a handler like this
  -- handler to remove the styling when the control loses focus
  set the textStyle of the target to empty
end focusPOut
```

Comments

Handle the focusOut message if you want to perform preparation or do other tasks when a control loses the keyboard focus.

Comments:

If the control is an unlocked field or a button whose menuMode is "comboBox", the closeField or exitField message is sent to it instead of the focusOut message.

focusPattern

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

effective keyword, focus command, focusColor property, focusedObject function, focusIn message, focusOut message, patterns property, traversalOn property

Summary

Specifies the pattern used for an object's outline when it has the insertion point or is active (focused).

Syntax

set the focusPattern of object to {patternNumber | imageID | empty}

Example Code

```
set the focusPattern of this stack to 2215
```

Comments

Use the focusPattern property to specify the pattern used for the outline around an active (focused) control.

Value:

The focusPattern of an object is a pattern specifier.

A patternNumber is a built-in pattern number between 1 and 164. (These patterns correspond to Revolution's built-in patterns 136 to 300.)

An imageID is the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the focusPattern for all objects is empty.

Comments:

Pattern images can be color or black-and-white.

Cross-platform note: To be used as a pattern on Mac OS systems, an image must be 128x128 pixels or less, and both its height and width must be a power of 2. To be used on Windows and Unix systems,

height and width must be divisible by 8. To be used as a fully cross-platform pattern, both an image's dimensions should be one of 8, 16, 32, 64, or 128.

The focusPattern of controls is drawn starting at the control's upper right corner: if the control is moved, the pattern does not shift.

Setting the focusPattern of an object to empty allows the focusPattern of the object's owner to show through. Use the effective keyword to find out what color is used for the object, even if its own focusPattern is empty.

The setting of the focusPattern property has different effects, depending on the object type:

- The focusPattern of a stack, card, or group determines the focusPattern of each object in the stack, card, or group that does not have its own focusPattern.

- The focusPattern of a button is used to outline the button when it is focused. If the button's style is menu, or if the button's traversalOn property is false, the focusPattern has no effect.

- The focusPattern of a field or scrollbar is used to outline the object when it is focused. If the object's traversalOn property is false, the focusPattern has no effect.

- The focusPattern of a graphic, image, player, audioClip, videoClip, or EPS object has no effect.

If an object's focusPattern is set, the pattern is shown instead of the color specified by the focusColor.

The focusPattern property has no effect if the lookAndFeel property is set to Macintosh.

focusPixel

property

Synonyms

eighthPixel

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundPixel property, borderPixel property, bottomPixel property, colorMap property, focusColor property, foregroundPixel property, hilitePixel property, screenColors function, shadowPixel property, topPixel property, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies which entry in the color table is used for the color of an object's outline when it has the insertion point or is active (focused).

Syntax

set the focusPixel of object to colorNumber

Example Code

```
set the focusPixel of this stack to 33
```

Comments

Use the focusPixel property to change the focus color of an object when the bit depth of the screen is 8 bits (256 colors) or less.

Value:

The focusPixel of an object is an integer between zero and (the screenColors - 1). It designates an entry in the current color table.

By default, the focusPixel for all objects is empty.

Comments:

The focusPixel property specifies which entry in the color table is used for an object's focus color. It is similar to the focusColor property, but is specified as an entry in the current color table, rather than as a color reference.

The color table can be set by changing the colorMap property.

folders

function

Synonyms

directories

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

answer folder command, defaultFolder property, files function, rename command, revCopyFolder command, revDeleteFolder command, revMoveFolder command, specialFolderPath function, About filename specifications and file paths, How to change the current folder used for file operations, Recipe for checking whether a volume or folder is write-protected

Summary

Returns a list of the folders in the current defaultFolder.

Syntax

the [detailed | long] folders

folders()

Example Code

```
set the defaultFolder to line x of the folders
put the detailed folders into folderList
```

Comments

Use the folders function to obtain a list of folders to display for the user, or to perform an action on each folder in the current folder.

Value:

The folders function returns a list of folder names, one per line.

The detailed folders form returns a list of folders, one file per line. Each line contains the following attributes, separated by commas. (Several attributes are used only for compatibility with the files function and do not have any meaning for folders.)

- ï The folder's name
- ï The folder's size in bytes (always zero)
- ï The folder's resource fork size in bytes (always zero)
- ï The folder's creation date in seconds (Mac OS, OS X, and Windows systems only)
- ï The folder's modification date in seconds
- ï The folder's last-accessed date in seconds (Unix and Windows systems only)

- ï The folder's last-backup date in seconds (Mac OS and OS X systems only)
- ï The folder's owner (Unix systems only)
- ï The folder's group owner (Unix systems only)
- ï The folder's access permissions
- ï The folder's creator and file type (always "????????") (Mac OS and OS X only)

Any attribute that is not supported on the current system is reported as "0" (zero).

The first line is always "..", representing the folder that contains the current folder.

Comments:

Files in the defaultFolder are not included. To get a list of folders, use the files function.

The names of aliases (on Mac OS systems), symbolic links (on Unix systems), and shortcuts (on Windows systems) are included in the value returned by the folders if they link to a folder. If they link to a file, they are not included.

This function lists only the folders on the top level of the defaultFolder. To get a list of subfolders within a folder, set the defaultFolder to that folder and then use the folders function again.

When listed in the detailed folders form, each folder's name is URL-encoded. To obtain the name in plain text, use the URLDecode function. If the detailed modifier is not used, the folder name is not encoded.

The access permissions returned in the detailed files form consist of three octal digits. The form is the same as that used for the umask command.

Changes to Transcript:

The detailed folders form was introduced in version 1.1. In previous versions, the folders function provided only a list of folder names.

fontLanguage

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

fontNames function, uniDecode function, uniEncode function, How to enter or display Unicode text in a field

Summary

Returns the language associated with a Unicode font.

Syntax

the fontLanguage of fontName

fontLanguage(fontName)

Example Code

```
the fontLanguage of "Osaka"
```

Comments

Use the fontLanguage function to list only the fonts intended for a particular language.

Parameters:

The fontName is the name of any font that is installed on the current system.

Value:

The fontLanguage of a font is one of the following:

• ANSI (for English-language fonts)

• Arabic

• Bulgarian

• Chinese

• Greek

• Hebrew

• Japanese

• Korean

• Polish

• Roman

• Russian

- Thai
- Turkish
- SimpleChinese
- Ukrainian

Comments:

For a list of all fonts installed on the current system, use the `fontNames` function.

Cross-platform note: On Unix systems, the `fontLanguage` always returns "English".

fontNameNames

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

fontLanguage function, fontSizes function, fontStyles function, printTextFont property, scriptTextFont property, textFont property, Recipe for creating a font list for a menu

Summary

Returns a list of the currently installed fonts.

Syntax

the fontNames

fontNameNames([printer])

Example Code

```
the fontNames
fontNameNames()
fontNameNames(printer)
if "Monaco" is among the lines of the fontNames
    then set the textFont of me to "Monaco"
```

Comments

Use the fontNames function to find out whether a particular font is available before using it, or to display a list of fonts.

Value:

The fontNames function returns a list of font names, one per line.

Comments:

The fontNames(printer) form returns the names of fonts available on the currently selected printer. Use this form when printing on a printer with its own resident fonts (such as a Postscript printer) to ensure that the fonts you're using are available on the printer.

If you don't specify "printer", the fonts installed on the system and available to the application for screen display are listed.

Changes to Transcript:

The fontNames(printer) form was introduced in version 2.0.

fontSizes

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

fontNames function, fontStyles function, printTextSize property, scriptTextSize property, textSize property

Summary

Returns a list of the font sizes available for a specified font.

Syntax

the fontSizes of fontName

fontSizes(fontName)

Example Code

```
the fontSizes of "Arial"
```

Comments

Use the fontSizes function to determine which font sizes can be displayed or printed without creating "jaggies".

Parameters:

The fontName is the name of a font that is installed on the system.

Value:

The fontSizes function returns a list of numbers that represent point sizes, one per line.

Comments:

If zero appears in the list returned by the fontSizes function, the font is scalable, meaning that it can be displayed at any size supported by the operating system.

fontStyles

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

fontNames function, fontSizes function, printTextStyle property, textStyle property

Summary

Returns a list of styles available for a font and size.

Syntax

fontStyles(fontName,fontSize)

Example Code

```
fontStyles("Helvetica",9)
fontStyles(thisFont,0)
```

Comments

Use the fontStyles function to determine which font styles can be displayed or printed.

Parameters:

The fontName is the name of a font that is installed on the system.

The fontSize is a point size that exists for the font.

Value:

The fontStyles function returns a list of available styles, one per line.

Comments:

If you specify a fontSize of zero, the styles returned are the ones that are scalable for the specified font.

for
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

forever keyword, repeat control structure, until keyword, while keyword

Summary

Used in a repeat control structure to specify the number of times the loop should repeat.

Syntax

Example Code

```
repeat for 10 times  
repeat for (the number of buttons)
```

Comments

Use the for keyword to specify a fixed number of iterations of a loop.

Comments:

The number may be a literal number, or an expression that evaluates to a number. (If the number is not an integer, it is rounded to the nearest integer.) Either way, the number is evaluated when the repeat control structure is entered, and is not re-evaluated as a result of statements in the loop. For example, if the repeat control structure begins with the line

```
repeat for the number of cards
```

then this number is evaluated as of the start of the loop, and the number does not change even if the statements in the loop create or delete cards. For this reason, use caution when using the for keyword with an expression: do not assume the expression will have its current value when the statements inside the loop are executing.

foregroundColor

property

Synonyms

foreColor, firstColor, textColor, thumbColor, fillFore, penFore

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundColor property, borderColor property, bottomColor property, colorNames function, colors property, effective keyword, focusColor property, foregroundPattern property, hiliteColor property, linkColor property, owner property, shadowColor property, topColor property, About colors and color references, Color Names Reference, How to store styled text in a variable or property, Recipe for setting the red channel of an object, Recipe for translating a color name to an RGB numeric triplet, Text menu > Color, Shortcut to remove font changes from text

Summary

Specifies the color of object text and borders.

Syntax

set the foregroundColor of object to {empty | colorName | RGBColor}

set the foregroundColor of [chunk of] field to {empty|colorName|RGBColor}

Example Code

```
set the foregroundColor of button "Help!" to "red"
set the foregroundColor of this card to "#FF00FF"
set the foregroundColor of the mouseChunk to 0,255,0
```

Comments

Use the foregroundColor property to change the color of text or the color that fills an object.

Value:

The foregroundColor of an object is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the foregroundColor for all objects is empty.

Comments:

Setting the foregroundColor of an object to empty allows the foregroundColor of the object's owner to show through. Use the effective keyword to find out what color is used for the object, even if its own foregroundColor is empty.

The setting of the foregroundColor property has different effects, depending on the object type:

- ï The foregroundColor of a stack, card, or group determines the foregroundColor of each object in the stack, card, or group that does not have its own foregroundColor.
- ï The foregroundColor of a button is used for the text of the button. If the button's showName property is false, the foregroundColor has no effect.
- ï The foregroundColor of a field determines the color of the field's text. If you set the foregroundColor of a chunk of a field, only that chunk is affected. If a chunk of text contains text of more than one color, the foregroundColor of that chunk reports "mixed".

The foregroundColor also determines the color of the blinking insertion point when it is in the field.

- ï The foregroundColor of a scrollbar determines the text color used to show the value of the scrollbar's current position. If the scrollbar's showValue property is false, the foregroundColor has no effect.
- ï The foregroundColor of a graphic determines the color of the graphic's outline. (The graphic's border is outside the outline. By default, the showBorder property of newly-created graphics is set to false, so this border is not visible.)
- ï The foregroundColor of a player, audio clip, video clip, or EPS object has no effect.
- ï The foregroundColor of an image is the first color in the image's color palette.

If an object's foregroundPattern is set, the pattern is shown instead of the color specified by foregroundColor.

foregroundPattern

property

Synonyms

forePattern, textPattern, thumbPattern

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backgroundPattern property, borderPattern property, bottomPattern property, focusPattern property, foregroundColor property, hilitePattern property, patterns property, shadowPattern property, topPattern property

Summary

Specifies the pattern used for object text and borders.

Syntax

set the foregroundPattern of object to {patternNumber | imageID | empty}

set the foregroundPattern of [chunk of] field to

{patternNumber | imageID | empty}

Example Code

```
set the foregroundPattern of me to 1034
```

```
set the textPattern of line 2 of field "Cards" to 111
```

Comments

Use the foregroundPattern property to change the pattern used for text or the pattern that fills an object.

Value:

The foregroundPattern of an object is a pattern specifier.

A patternNumber is a built-in pattern number between 1 and 164. (These patterns correspond to Revolution's built-in patterns 136 to 300.)

An imageID is the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the foregroundPattern for all objects is empty.

Comments:

Pattern images can be color or black-and-white.

Cross-platform note: To be used as a pattern on Mac OS systems, an image must be 128x128 pixels or less, and both its height and width must be a power of 2. To be used on Windows and Unix systems, height and width must be divisible by 8. To be used as a fully cross-platform pattern, both an image's dimensions should be one of 8, 16, 32, 64, or 128.

The foregroundPattern of controls is drawn starting at the control's upper right corner: if the control is moved, the pattern does not shift.

Setting the foregroundPattern of an object to empty allows the foregroundPattern of the object's owner to show through. Use the effective keyword to find out what pattern is used for the object, even if its own foregroundPattern is empty.

The setting of the foregroundPattern property has different effects, depending on the object type:

- The foregroundPattern of a stack, card, or group determines the foregroundPattern of each object in the stack, card, or group that does not have its own foregroundPattern.

- On Unix systems, the foregroundPattern of a button is used for the text of the button. If the button's showName property is false, the foregroundPattern has no effect.

- On Unix systems, the foregroundPattern of a field determines the color of the field's text. If you set the foregroundColor of a chunk of a field, only that chunk is affected. The foregroundPattern also determines the color of the blinking insertion point when it is in the field.

- On Unix systems, the foregroundPattern of a scrollbar determines the pattern of the text used to show the value of the scrollbar's current position. If the scrollbar's showValue property is false, the foregroundPattern has no effect.

- The foregroundPattern of a graphic determines the pattern used for the graphic's outline. (The borderPattern determines the pattern used for the graphic's border, which is outside the outline.)

- The foregroundPattern of a player, image, audio clip, video clip, or EPS object has no effect.

If an object's foregroundPattern is set, the pattern is shown instead of the color specified by foregroundColor.

Cross-platform note: On Mac OS and Windows systems, the foregroundPattern applies only to object borders, not to text; text cannot be drawn with a pattern, only with a color.

foregroundPixel

property

Synonyms

forePixel, firstPixel

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundPixel property, borderPixel property, bottomPixel property, colorMap property, focusPixel property, foregroundColor property, hilitePixel property, screenColors function, shadowPixel property, topPixel property, Recipe for setting the red channel of an object, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies which entry in the color table is used for an object's text and borders.

Syntax

set the foregroundPixel of object to colorNumber

Example Code

```
set the foregroundPixel of this stack to 0
```

Comments

Use the foregroundPixel property to change the foreground color of an object when the bit depth of the screen is 8 bits (256 colors) or less.

Value:

The foregroundPixel of an object is an integer between zero and (the screenColors - 1). It designates an entry in the current color table.

By default, the foregroundPixel for all objects is empty.

Comments:

The foregroundPixel property specifies which entry in the color table is used for an object's foreground color. It is similar to the foregroundColor property, but is specified as an entry in the current color table, rather than as a color reference.

The color table can be set by changing the colorMap property.

forever

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

for keyword, repeat control structure, until keyword, while keyword

Summary

Used in a repeat loop structure to specify that the loop should repeat until halted by an exit repeat.

Syntax

Example Code

```
repeat forever
```

Comments

Use the forever keyword to repeat until an exit repeat statement within the loop explicitly ends the loop or until the heat death of the universe, whichever comes first.

Comments:

All forms of the repeat control structure that test for a condition, test at the top of the loop (before each iteration). If you want to test at the bottom of the loop, use the forever keyword, and use a conditional exit repeat at the bottom of the loop:

```
repeat forever
  -- statements here
  if myCondition is true then exit repeat
end repeat
```

format

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

baseConvert function, binaryEncode function, charToNum function, numberFormat property, numToChar function

Summary

Returns a formatted string that has been transformed according to the rules of the C "printf()" function.

Syntax

`format(baseString[,valuesList])`

Example Code

```
format("Hello world") -- returns "Hello world"
format("Hello") -- returns "Hello" on one line, "world" on next
format("%1.3e",865.3) -- returns 8.653e+02, scientific notation
format("%o in octal%x in hex.",myNumber,myNumber)
format("%45d",5) -- returns "5" preceded by 45 spaces
```

Comments

Use the format function to create strings in special formats.

Parameters:

The baseString is a string that can include one or more special formatting incantations.

The valuesList is a list of strings (or expressions that evaluate to strings), separated by commas. The valuesList contains as many values as there are incantations in the baseString.

Value:

The format function returns a string.

Comments:

The format function works by taking a baseString that contains formatting incantations, transforming each member of the valuesList according to the corresponding incantation, then substituting the transformed string for the incantation in the baseString. It also transforms C escape sequences in the baseString into their single-character equivalents.

The valid incantations are as follows:

String:

`%[charLength]s`

The corresponding value is unchanged, except that if a `charLength` is specified, if the string is shorter than the `charLength`, enough leading spaces are added to make it `charLength` characters long. If the length of the string is equal to or greater than the `charLength`, it is unchanged. For example, the incantation `%3s` transforms "H" to " H".

Character:

`%[charLength]c`

The corresponding value is treated as an ASCII value and translated to the corresponding character. If a `charLength` is specified, one fewer leading spaces are added to make it `charLength` characters long. For example, the incantation `%2c` transforms 65 to " A" (65 is the ASCII value of the character "A").

Decimal number:

`%[charLength]d`

The corresponding value is rounded to the nearest integer: if the fractional part is .5 or more, the value is rounded up, and otherwise it is rounded down. (If the value is negative, if the fractional part is .5, the value is rounded down.) If a `charLength` is specified, if the length of the resulting number is less than the `charLength`, enough leading spaces are added to make it `charLength` characters long. If the length of the resulting number is equal to or greater than the `charLength`, it is unchanged. For example, the incantation `%2d` transforms "2.3" to " 2".

Unsigned integer:

`%[charLength]u`

Like `%[charLength]d`, except that if the value is negative, it is subtracted from the largest long integer allowed by the current operating system. (On most operating systems, this is 2^{32} , or 4,294,967,296.)

Octal:

`%[charLength]o`

The corresponding value is assumed to be a decimal number, rounded to the nearest integer, then converted to an octal number. If a `charLength` is specified, if the length of the resulting number is less than the `charLength`, enough leading spaces are added to make it `charLength` characters long. If the length of the resulting number is equal to or greater than the `charLength`, it is unchanged. For example, the incantation `%3o` transforms "10.7" to " 13".

Hexadecimal:

`%[charLength]x`

The corresponding value is assumed to be a decimal number, rounded to the nearest integer, then converted to a hexadecimal number. If a `charLength` is specified, if the length of the resulting number is less than the `charLength`, enough leading spaces are added to make it `charLength` characters long. If the length of the resulting number is equal to or greater than the `charLength`, it is unchanged. For example, the incantation `%4x` transforms "30.3" to " 1e".

`%[charLength]X`

Like `%[charLength]x`, except that the hex digits A-F are given in uppercase. For example, the incantation `%4x` transforms "30.3" to " 1E".

Floating-point:

`%[charLength].[precision]f`

The corresponding value is a real number. If a precision is specified, if the number of digits after the decimal point is greater than the precision, the number is rounded to the specified number of digits after the decimal point. If the number of digits is less than the precision, enough trailing zeroes are added to make precision digits. If no precision is specified, the number is formatted to six decimal places. If a charLength is specified, if the total length of the resulting number is less than the charLength, enough leading spaces are added to make it charLength characters long; if the length of the resulting number is equal to or greater than the charLength, it is unchanged.

`%[charLength.precision]g`

Like `%[charLength].[precision]f`, except that trailing zeroes are not added if the number of digits is less than the precision.

Scientific notation:

`%[charLength.precision]e`

The corresponding value is a real number. First the number is transformed to scientific notation: expressed as a number between 1 and 10 (or -10 and 1), multiplied by the appropriate power of 10. If a precision is specified, if the number of digits after the decimal point is greater than the precision, the number is rounded to the specified number of digits after the decimal point. If the number of digits is less than the precision, enough trailing zeroes are added to make precision digits. If no precision is specified, the number is given to six digits after the decimal point. If a charLength is specified, if the total length of the resulting number is less than the charLength, enough leading spaces are added to make it charLength characters long; if the length of the resulting number is equal to or greater than the charLength, it is unchanged. For example, the incantation `%3.2e` transforms "232.4" to "2.32e+02".

`%[charLength.precision]E`

Like `%[charLength.precision]e`, except that the "E" separating the number from the power of ten is uppercase. For example, the incantation `%3.2e` transforms "232.4" to "2.32E+02".

Note: If a zero is included immediately before the charLength parameter in any formatting incantation that allows padding, the resulting value is padded (if necessary) with zeroes instead of spaces. For example, the incantation `%03s` transforms "H" to "00H".

If any of the following C escape sequences are present in the baseString, the format function transforms them to the equivalent character:

Control-G (bell)

Control-H (backspace)

Control-L (formfeed)

Control-J (linefeed)

Control-M (return)

Control-I (tab)

formatForPrinting

property

Synonyms

Objects

stack

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer printer command, print command, textFont property

Summary

Specifies whether font layout is done using printer fonts or screen fonts.

Syntax

set the formatForPrinting of stack to {true | false}

Example Code

```
set the formatForPrinting of stack "Print Template" to true
```

Comments

Use the formatForPrinting property to improve printout appearance on Windows systems.

Value:

The formatForPrinting of a stack is true or false.

By default, the formatForPrinting property of newly created stacks is set to false.

Comments:

Windows systems may use different font versions for printing and for screen display, and the spacing of the print version may differ from the spacing of the screen version. This can result in layouts and line breaks differing between the screen display and the printed result. For the best appearance of printed cards in a stack, make sure the stack is closed (and not in memory), then set the stack's formatForPrinting property to true before opening the stack to print it.

Important! Do not edit field text in a stack whose formatForPrinting is true. Doing so can cause display anomalies. Set the formatForPrinting property to false before you make changes to text in fields.

The spacing of printer font versions usually results in a difficult-to-read display when these fonts are used for screen viewing. To avoid display problems, set the formatForPrinting property to true only when printing. To let the user preview the appearance of the printed output, set the formatForPrinting property to true before opening the stack.

Important! Fonts inherited from another stack are not updated when you set the `formatForPrinting` of a stack. If the stack will be printed, make sure that either the stack's `textFont` property is set to a font name (not set to empty), or all fields to be printed have their own font rather than inheriting it.

If the stack's `formatForPrinting` property is true, the setting of the `windowBoundingRect` property is ignored when the stack is opened or maximized.

formattedHeight

property

Synonyms

Objects

button, field, image, player, group, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

dontWrap property, formattedRect property, formattedTop property, formattedWidth property, height property, lockLocation property, pageHeights property, textHeight property, textHeightSum function, Recipe for adding a scrollbar to a field when the contents overflow it, Recipe for a "roll credits" effect

Summary

Reports the height needed by an object to display its full contents without scrolling.

Syntax

get the formattedHeight of object

get the formattedHeight of [chunk of] field

Example Code

```
set the height of field 1 to the formattedHeight of field 1
put the formattedHeight of the clickLine into selectedHeight
get the formattedHeight of word 1 to 17 of field "Info"
```

Comments

Use the formattedHeight property to determine how much vertical space an object needs. For example, if your stack has a field whose contents change for each card, use the field's formattedHeight property in an openCard handler to resize the field for its contents on each card.

Value:

The formattedHeight of an object is a positive integer. The object must be on the current card of an open stack.

This property is read-only and cannot be set.

Comments:

If you specify a card or group, the formattedHeight reports the height of a rectangle that includes all objects in that card or group whose visible property is true.

If you specify an image or player, the `formattedHeight` property reports the original un-scaled height of the image or movie.

If you specify an object in a group, the value reported is the `formattedHeight` that object requires for the current card, so if you want to get the `formattedHeight` of a field's text on a certain card, you must go to that card first.

The `formattedHeight` of a chunk in a field is the amount of vertical space that portion of the field's text requires, taking line breaks into account.

formattedLeft

property

Synonyms

Objects

field, group, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

formattedRect property, formattedWidth property, left property, resizeStack message, right property

Summary

Reports the distance between the left edge of the stack window and the leftmost object in a card or group.

Syntax

get the formattedLeft of {group | card}

get the formattedLeft of chunk of field

Example Code

```
get the formattedLeft of this card
```

```
put the formattedLeft of the clickChunk into clickedHeight
```

Comments

Use the formattedLeft property to find the boundary of a group or chunk, or to determine how far to scroll horizontally to bring a chunk of text into the viewable area of a field.

Value:

The formattedLeft of a group or card is an integer.

This property is read-only and cannot be set.

Comments:

If you specify a card or group, the formattedLeft reports the distance in pixels between the left edge of the stack window and the left edge of the leftmost object in the card or group. (Objects whose visible property is false are ignored.)

If the leftmost object is to the left of the stack window's left edge, the formattedLeft is a negative number.

The formattedLeft of a chunk in a field is the distance from the left edge of the stack window to the left edge of an imaginary box containing the text in the chunk.

formattedRect

property

Synonyms

Objects

group, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

formattedHeight property, formattedLeft property, formattedWidth property, rectangle property

Summary

Reports the rectangle that surrounds all the objects in a card or group.

Syntax

get the formattedRect of {card | group}

get the formattedRect of chunk of field

Example Code

```
put the formattedRect of group "Nav Buttons" into rectToSelect
```

Comments

Use the formattedRect property to find the boundary of a group or chunk, or to determine the screen location of a chunk of text.

Value:

The formattedRect of a group or card consists of four integers, separated by commas.

This property is read-only and cannot be set.

Comments:

If you specify a card or group, the formattedRect reports the smallest rectangle that encloses all the object in that group or card. (Objects whose visible property is false are ignored.) The four items in the rectangle are:

- 1: horizontal distance from the left edge of the stack to the left edge of the rectangle
- 2: vertical distance from the top edge of the stack to the top edge of the rectangle
- 3: horizontal distance from the left edge of the stack to the right edge of the rectangle
- 4: vertical distance from the left edge of the stack to the bottom edge of the rectangle

The formattedRect of a chunk in a field is the smallest rectangle that encloses the entire chunk.

formattedText

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

dontWrap property, htmlText property, return constant, text property, Recipe for truncating text to a pixel width, Recipe for word-wrapping text to a line length

Summary

Unwraps hard-wrapped lines, or provides the hard-wrapped contents of a field.

Syntax

set the formattedText of field to string

get the formattedText of field

Example Code

```
write the formattedText of field 1 to file myTextFile
```

Comments

Use the formattedText property to convert between unwrapped text (suitable for use in most programs) and wrapped text (suitable for use in programs that require fixed line lengths, such as some email systems).

Value:

The formattedText of a field is a string.

Comments:

When you get a field's formattedText, the field's text is converted to hard-wrapped text: An end-of-line character is inserted at the end of each screen line, and wherever a return character occurs in the field, two end-of-line characters are substituted. (If the field's dontWrap is true, each screen line ends in an end-of-line character, so two end-of-line characters are substituted for each return character in the field.)

Cross-platform note: The end-of-line character depends on the platform: a return (ASCII 13) on Mac OS systems, a linefeed (ASCII 10) on Unix systems, or a return and linefeed on Windows systems.

When you set a field's `formattedText` property, the string is unwrapped before being put in the field. Double end-of-line characters are converted to a single end-of-line character, and single end-of-line characters are converted to spaces.

formattedTop

property

Synonyms

Objects

field, group, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

formattedHeight property, formattedLeft property, formattedRect property, height property, pageHeights property, resizeStack message, top property

Summary

Reports the distance between the top edge of the stack window and the topmost object in a card or group.

Syntax

get the formattedTop of {card | group}

get the formattedTop of chunk of field

Example Code

```
get the formattedTop of card "Preferences"  
set the scroll of field 1 to the formattedTop of line 30 of field 1
```

Comments

Use the formattedTop property to find the boundary of a group or chunk, or to determine how far to scroll vertically to bring a chunk of text into the viewable area of a field.

Value:

The formattedTop of a group or card is an integer.

This property is read-only and cannot be set.

Comments:

If you specify a card or group, the formattedTop reports the distance in pixels between the top edge of the stack window and the top edge of the uppermost object in the card or group. (Objects whose visible property is false are ignored.)

If the uppermost object is above the top of the stack window, the formattedTop is a negative number.

The formattedTop of a chunk in a field is the distance from the top of the stack to the top of an imaginary box containing the text in the chunk.

formattedWidth

property

Synonyms

Objects

button, field, image, player, group, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

dontWrap property, formattedHeight property, formattedLeft property, formattedRect property, lockLocation property, width property, Recipe for truncating text to a pixel width

Summary

Reports the width needed by an object to display its full contents without scrolling.

Syntax

get the formattedWidth of object

get the formattedWidth of [chunk of] field

Example Code

```
if the formattedWidth of this card > the width of this card then beep
```

Comments

Use the formattedWidth property to adjust an object's size according to the space needed to display its contents.

Value:

The formattedWidth of an object is a positive integer. The object must be on the current card of an open stack.

This property is read-only and cannot be set.

Comments:

If you specify a card or group, the formattedWidth reports the width of a rectangle that includes all objects in that card or group whose visible property is true.

If you specify an image or player, the formattedWidth property reports the original un-scaled width of the image or movie.

If you specify a field, the formattedWidth reports the width required by the field's text. If the field's dontWrap property is set to true, the formattedWidth reports the total width of the text in the field. If the

`dontWrap` is false, the `formattedWidth` reports the minimum width required to keep the current line breaks.

If you specify an object in a group, the value reported is the `formattedWidth` that object requires for the current card, so if you want to get the `formattedWidth` of a field's text on a certain card, you must go to that card first.

The `formattedWidth` of a chunk in a field is the amount of horizontal space that portion of the field's text requires, taking line breaks into account.

formfeed

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

break keyword, constant command, return constant

Summary

Equivalent to the form feed character (ASCII 12, Control-L).

Syntax

Example Code

```
put formFeed after collatedData
```

Comments

Use the formFeed constant as an easier-to-read substitute for numToChar(12).

Comments:

The formfeed constant is needed because you can't type the character it represents in a script.

The form feed character is used by some applications and printers to indicate a page break.

foundChunk

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickChunk function, find command, foundField function, foundLine function, foundLoc function, foundText function, mouseChunk function, selectedChunk function, How to remove the box around found text

Summary

Returns a chunk expression describing the location of the text that was found by the most recent find command.

Syntax

the foundChunk

foundChunk()

Example Code

```
the foundChunk
if the foundChunk is not word 1 of thingToFind then find thingToFind
```

Comments

Use the foundChunk function after a find command to determine where the text was found.

Value:

The foundChunk function returns a chunk expression of the form char startChar to endChar of field fieldNumber.

Comments:

The foundChunk function is cleared when the text selection moves into the foundField or when the current card is closed. At the same time, the box the find command draws around the found text disappears. If there is no box, the foundChunk function returns empty.

The return value reports the text that was found: the startChar is the first character of the found text, and the endChar is the last character. For example, if the most recent find command was find "foo bar", either word might have been found.

To get the actual text that was found, use the `foundText` function.

foundField

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickField function, find command, foundChunk function, foundLine function, foundLoc function, foundText function, mouseChunk function, selectedField function, How to remove the box around found text

Summary

Returns the field in which text was located by the most recent find command.

Syntax

the foundField

foundField

Example Code

```
the foundField
if word 2 of the foundField is the number of field 1 then findItAgain
select the text of the foundField
```

Comments

Use the foundField function after a find command to determine where the text was found.

Value:

The foundField function returns the number property of the field.

Comments:

The foundField function is cleared when the text selection moves into the foundField or when the current card is closed. At the same time, the box the find command draws around the found text disappears. If there is no box, the foundField function returns empty.

foundLine

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickLine function, find command, foundChunk function, foundField function, foundLoc function, foundText function, mouseLine function, selectedLine function, How to remove the box around found text

Summary

Returns a chunk expression describing the line in which text was located by the most recent find command.

Syntax

the foundLine

foundLine()

Example Code

```
the foundLine
if value(the foundLine) contains "*" then answer value(the foundLine)
```

Comments

Use the foundLine function after a find command to determine where the text was found.

Value:

The foundLine function returns a chunk expression of the form line lineNumber of field fieldNumber.

Comments:

The foundLine function is cleared when the text selection moves into the foundField or when the current card is closed. At the same time, the box the find command draws around the found text disappears. If there is no box, the foundLine function returns empty.

To get a chunk expression describing the word or string that was found, use the foundChunk function.

foundLoc

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickLoc function, find command, foundChunk function, foundField function, foundLine function, foundText function, mouseLoc function, selectedChunk function, selectedLoc function, How to remove the box around found text

Summary

Returns the location of the top left corner of the box that surrounds text found by the find command.

Syntax

the foundLoc

foundLoc()

Example Code

```
the foundLoc
drag from the foundLoc to the bottomLeft of this card
```

Comments

Use the foundLoc function to determine the location in the stack window where the found text is shown.

Value:

The foundLoc function returns two positive integers separated by a comma.

Comments:

When the find command finds text, it surrounds the text with a box. The first item of the return value is the horizontal distance in pixels from the left edge of the stack to the left edge of this box. The second item of the returned value is the vertical distance from the top edge of the stack window to the top of the box.

The foundLoc function is cleared when the text selection moves into the foundField or when the current card is closed. At the same time, the box the find command draws around the found text disappears. If there is no box, the foundLoc function returns empty.

foundText

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickText function, find command, foundChunk function, foundField function, foundLine function, foundLoc function, mouseText function, selectedText function, How to remove the box around found text

Summary

Returns the text located by the most recent find command.

Syntax

the foundText

foundText()

Example Code

```
the foundText
put the foundText into whichWord
```

Comments

Use the foundText function to determine which word in the search string was found, and whether it matches a whole word or a portion of a word.

Value:

The foundText function returns the text that was found.

Comments:

Depending on what form of the find command you used, the text found may be different from the text you searched for. For example, the command

```
find "hurl"
```

can find any word that starts with the string "hurl", such as "hurling" or "hurler". In this case, the entire word—not just the portion specified in the find command—is surrounded by a box, and the foundText returns the entire word.

The `foundText` function is cleared when the text selection moves into the `foundField` or when the current card is closed. At the same time, the box the find command draws around the found text disappears. If there is no box, the `foundText` function returns empty.

To get the location of the text that was found, use the `foundChunk` function.

four
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, fourth keyword

Summary

Equivalent to the number 4.

Syntax

Example Code

```
divide quarterlyRevenue by four -- same as "...by 4"
```

Comments

Use the four constant when it is easier to read in a handler than the numeral 4.

fourth
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

four constant, last keyword, middle keyword, number property

Summary

Designates the fourth member of a set.

Syntax

Example Code

```
select after text of fourth field  
put empty into the fourth char of thisName
```

Comments

Use the fourth keyword in an object reference or chunk expression.

Comments:

The fourth keyword can be used to specify any object whose number property is 4. It can also be used to designate the fourth chunk in a chunk expression.

The word the is optional when using the fourth keyword.

frameCount

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constantMask property, currentFrame property, filename property, palindromeFrames property, repeatCount property

Summary

Reports the number of frames in an animated GIF image.

Syntax

get the frameCount of image

Example Code

```
put the frameCount of image "Welcome Animation" into estNumberOfSecs
```

Comments

Use the frameCount property to determine the length of a GIF animation.

Value:

The frameCount of an image is a non-negative integer.

This property is read-only and cannot be set.

Comments:

If the image is not an animated GIF, its frameCount is zero.

frameRate

property

Synonyms

Objects

videoClip

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

dontRefresh property, frameCount property, play command, scale property

Summary

Specifies the delay between frames when playing a video clip.

Syntax

set the frameRate of videoClip to timeDelay

Example Code

```
set the frameRate of videoClip 1 to 100
```

Comments

Use the frameRate property to speed up or slow down a movie.

Value:

The frameRate of a video clip is a non-negative integer.

By default, the frameRate property of newly created video clips is set to zero.

Comments:

The frameRate is the number of milliseconds to wait between frames. If the frameRate is zero, Revolution uses the frame rate built into the movie.

The setting of this property has no effect on Windows systems.

freeSize

property

Synonyms

Objects

stack

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

diskSpace function, hasMemory function

Summary

Always reports zero and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

get the freeSize of stack

Example Code

Comments

In HyperCard, the freeSize property reports the amount of wasted space in a stack. In Revolution, this property always reports zero.

from
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
to keyword

Summary

Used with several commands, usually to designate an origin point.

Syntax

Example Code

```
convert field "Date" from long date to short date  
move the mouseControl from 22,100 to 44,200  
remove the selectedObject from this card  
subtract 33 from thisValue
```

Comments

Use the from keyword to complete a command that requires it.

Comments:

The from keyword is used with the convert, drag, import, import snapshot, move, print card, read from file, read from process, read from socket, remove, remove script, request, and subtract commands.

front

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

back keyword, frontScripts function, insert script command, remove script command

Summary

Used with the insert script and remove script commands to designate a location in the message path before the target object.

Syntax

Example Code

```
remove the script of stack "Math Routines" from front
```

Comments

Use the front keyword to designate a frontScript.

Comments:

When used with the insert script or remove script command, the front keyword designates a frontScript which is to be placed in the message path before the target object.

frontScripts

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backScripts function, insert script command, remove script command, scriptLimits function, stacksInUse property

Summary

Returns a list of objects that have been inserted into the message path before the target object.

Syntax

the frontScripts

frontScripts()

Example Code

```
the frontScripts
```

```
if myID is not among the lines of the frontScripts then insertMe
```

Comments

Use the frontScripts function to find out which scripts receive messages and function calls before the target object.

Value:

The frontScripts function returns a list of the long ID property of all objects that have been inserted into the front, one ID per line.

Comments:

A script inserted into the front with the insert script command receives messages before all objects in the message path.

This includes messages sent with the send command, so if you send a message to an object, the objects in the frontScripts receive that message before the target object does. If the scripts in the frontScripts do not use the pass control structure to pass on the message to the next object, the target object never receives the message.

If more than one object is in the frontScripts, their order in the message path is the same as their order in the list. For example, the first object in the frontScripts receives messages before the second object. This order is the reverse of the order in which the objects were added with the insert script command.

When using the development environment, you can place as many objects as you want in the frontScripts. When using a standalone application, the number of frontScripts that can be active at once is specified by the scriptLimits function, and is currently ten.

ftp

keyword

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete URL command, http keyword, libURLErrorData function, libURLftpCommand function, libURLftpUpload command, libURLSetFTPMode command, libURLSetFTPStopTime, libURLSetFTPListCommand command, load command, URL keyword, URLStatus keyword, About containers, variables, and sources of value, About using URLs, uploading, and downloading, How to create a directory on an FTP server, How to list the files in an FTP directory, How to remove a file from an FTP server, How to use a password for an FTP server, Why can't I upload a file?, Why don't URLs work in a standalone?, Why was a downloaded file corrupted?

Summary

Used as a URL type with such commands as put and get to designate a file or directory on an FTP server.

Syntax

Example Code

```
put URL "ftp://ftp.example.com/public/" into filesList
get URL "ftp://john:passwd@ftp.example.net:2121/picture.jpg"
put URL "ftp://files.example.org/file.txt" into URL "file:myFile.txt"
put field "Upload" into URL "ftp://me:secret@ftp.example.net/file.txt"
```

Comments

Use the ftp keyword to upload or download files to or from an Internet site.

Comments:

The URL scheme "ftp" indicates information located on an FTP server. An ftp URL consists of the following parts:

1. The string "ftp://"
2. An optional user name and password, separated by a colon (:) and followed by "@"
3. The name of the server
4. An optional port number preceded by a colon (:) and followed by "/"
5. The name and location of a file or directory, starting with a slash (/)

If you don't specify a port number, port 21 is used. (This is the standard port for FTP.)

Note: Most public FTP servers do not require a user name and password. For such servers, you need not specify any user name or password. If you don't specify a user name or password, Revolution adds the "anonymous" user name and a dummy password automatically, in accordance with the conventions for public FTP servers.

Important! If your user name or password contains any of the characters ":", "@", "/", ".", or "|", use the URLEncode function to safely encode the user name or password before putting them into the URL. The following example constructs a URL for a user whose password contains the "@" character:

```
put "jim" into userName
put "jsmith@example.org" into userPassword
put "ftp://" & userName & ":" & URLEncode(userPassword)
```

```
& "@ftp.example.com/title.txt" into fileURLToGet
get URL fileURLToGet
```

Here are some examples of valid ftp URLs:

```
ftp://ftp.example.org/directory/ -- list of files and folders in a directory
ftp://ftp.example.org/directory/file.exe -- a file on the server
ftp://user:password@ftp.example.org/myfile -- a file accessed by a password
ftp://ftp.example.com:3992/somefile -- using a nonstandard FTP port
```

An ftp URL is a container, and you can use the expression URL ftpURL in any statement where any other container type is used. When you get the value of an ftp URL, Revolution downloads the URL from the server. (If you have previously cached the URL with the load command, it fetches the URL from the cache.)

A URL that ends with a slash (/) designates a directory (rather than a file). An ftp URL to a directory evaluates to a listing of the directory's contents. To change the format of directory listings, use the libURLSetFTPListCommand command.

If an error occurs during transfer of the data, the error is placed in the result function. The first word returned by the result function is "error", followed (where appropriate) by the text of the error message returned by the FTP server, including the server response code.

FTP uploads and downloads that are performed using the ftp keyword are always transferred in binary mode: no character translation is performed. If you need to translate characters—for example, if you are uploading a text file to a different operating system and want to translate line endings—you must do so before uploading the file, since the put command will not do it for you.

Note: Downloading a URL by using it in an expression is a blocking operation: that is, the handler pauses until Revolution is finished getting the URL. Since contacting a server may take some time due to network lag, URL operations may take long enough to be noticeable, so you may want to set the cursor to the watch or otherwise indicate a delay to the user.

Important! If a blocking operation involving a URL (using the put command to upload a URL, the post command, the delete URL command, or a statement that gets an ftp or http URL) is going on, no other blocking URL operation can start until the previous one is finished. If you attempt to use a URL in an expression, or put data into a URL, while another blocking URL operation is in progress, the result is set to "Error Previous request not completed".

Downloading a URL does not prevent other messages from being sent during the download: the current handler is blocked during the download, but other handlers are not. This means that if, for example, your application has a button that downloads a URL, the user might click the button again (or click another button that downloads another URL) while the first URL is still being downloaded. In this case, the second download is not performed and the result is set to "Error Previous request not completed." To avoid this problem, you can set a flag while a URL is being downloaded, and check that flag when trying to download URLs to make sure that there is not already a current download in progress.

The following example shows how to set a flag in a global variable to prevent multiple downloads. The variable "downloadInProgress" is set to true while a download is going on, and back to false when the download concludes. If the user clicks the button again while the download is still going on, the handler simply beeps:

```
on mouseUp
  global downloadInProgress
  if downloadInProgress then
    beep
  exit mouseUp
end if
put true into downloadInProgress -- about to start
put URL (field "FTP URL to get") into field "Command Result"
put false into downloadInProgress -- finished
end mouseUp
```

To send any FTP command to an FTP server, use the libURLftpCommand function.

For technical information about URLs and the ftp URL scheme, see RFC 1630 at <http://www.ietf.org/rfc/rfc1630.txt>.

Important! The ftp keyword is part of the Internet library. To ensure that the keyword works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the ftp keyword is not yet available. This may affect attempts to use this keyword in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the ftp keyword can be used in any handler.

Changes to Transcript:

The ability to specify a port number was added in version 2.0. In previous versions, port 21 was always used for FTP transactions.

ftpProxy

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

httpProxy property

Summary

The ftpProxy property is not implemented and is reserved.

Syntax

Example Code

Comments

function

control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

end keyword, functionNames function, getProp control structure, on control structure, param function, params function, return control structure, About commands and functions, About the structure of a script, How to write your own commands and functions

Summary

Defines a custom function handler.

Syntax

```
function functionName [parametersList]
  statementList
end functionName
```

Example Code

Comments

Use the function control structure to implement a custom function.

Form:

The first line of a function handler consists of the word "function" followed by the function's name. If the function has any parameters, their names come after the function name, separated by commas.

The last line of a function handler consists of the word "end" followed by the function's name.

Parameters:

The functionName is a string up to 65,535 characters in length.

The parametersList consists of one or more parameter names, separated by commas.

The statementList consists of one or more Transcript statements.

Comments:

The purpose of a function is to compute a value and return it to the handler that called the function. The function's value is returned by a return control structure within the function handler.

A function handler can contain any set of Transcript statements. Most functions contain a return statement, which returns the value to the calling handler. This example of a custom function uses two parameters and returns a string:

```
function reversedName firstName,lastName
  -- firstName and lastName are parameters
  put lastName,firstName into constructedName
  return constructedName
end reversedName
```

You create a custom function by writing a function handler for it. When you use the function in a script, the function call is passed through the message path. When it reaches the object whose script contains the function handler, the statements in the handler are executed.

A custom function is called by name, just like a built-in function, as part of an expression. For example, this handler calls the custom function above:

```
on mouseUp
  ask "What is your first name?"
  put it into firstParam
  ask "What is your last name?"
  put it into secondParam
  put reversedName(firstParam,secondParam) into field "Name"
end mouseUp
```

A function can call itself. The following example calls itself to compute the factorial of an integer:

```
function factorial theNumber
  if theNumber <= 1 then return 1
  else return theNumber * factorial(theNumber -1)
end factorial
```

functionKey

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

arrowKey message, commandKeyDown message, keyDown message

Summary

Sent when the user presses a function key.

Syntax

functionKey keyNumber

Example Code

```
on functionKey theKey
  if theKey is 8 then showPaintTools -- a custom handler
  else pass functionKey
end functionKey
```

Comments

Handle the functionKey message to let the user perform some action with a function key.

Parameters:

The keyNumber is the number of the function key.

Comments:

The message is sent to the active (focused) control, or to the current card if no control is focused.

Keyboards typically have twelve function keys, so you can use numbers from 1 to 12 to implement custom actions.

functionNames

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

commandNames function, constantNames function, function control structure, propertyNames function, variableNames function

Summary

Returns a list of all built-in functions in Transcript.

Syntax

the functionNames

functionNames()

Example Code

```
the functionNames
if myHandlerName is in the functions then answer "Choose another name."
```

Comments

Use the functionNames function to check whether a particular function already exists in Transcript, to avoid using a reserved word for your own custom function handlers.

Value:

The functionNames function returns one function name per line.

Comments:

The functionNames function returns all the built-in functions that can be used in Transcript, including synonyms.

get
command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

getProp control structure, it keyword, put command, set command

Summary

Places the value of an expression in the it variable.

Syntax

get expression

Example Code

```
get the colors
get 2 * myRadius * pi -- gets area of circle
get message -- evaluates expression in message box
get URL "http://www.example.org/index.html"
```

Comments

Use the get command to fetch the value of an expression.

Parameters:po

The expression is any expression that yields a value.

Comments:

The get command is a shorthand way of writing the following statement:

put expression into it

getProp

control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

customKeys property, end keyword, function control structure, properties property, propertyName property, return control structure, setProp control structure, About custom properties and custom property sets, About the structure of a script

Summary

Handles the message sent to an object when you access one of its custom properties.

Syntax

```
getProp propertyName  
    statementList  
end propertyName
```

Example Code

Comments

Use the getProp control structure to perform a transformation on a custom property before its value is returned to the handler that requested it, or to implement a virtual property (one that is implemented only in a handler).

Form:

The first line of a getProp handler consists of the word "getProp" followed by the property's name.

The last line of a getProp handler consists of the word "end" followed by the property's name.

Parameters:

The propertyName is the name of a custom property.

The statementList consists of one or more Transcript statements, and can also include if, switch, try, or repeat control structures.

Comments:

The property's value is returned to the calling handler by a return control structure within the `getProp` handler. (A `getProp` handler works much like a custom function handler, in the sense that it exists to return a value.)

The `getProp` call passes through the message path, so a `getProp` handler for an object can be located in the object's script or in the script of any object further in the message path. For example, a `getProp` handler for a card property may be located in the script of the stack that the card belongs to.

If you use a custom property of an object within a `getProp` control structure for the custom property in the object's own script, no `getProp` call is sent to the object. (This is to avoid runaway recursion, where the `getProp` handler calls itself.) This is only the case for the custom property that the current `getProp` handler applies to. Setting a different custom property does send a `getProp` call. So does setting the same custom property for an object other than the one whose script contains the `getProp` handler.

Caution! If a `getProp` handler in one object's script uses the value of the custom property for a different object, and the first object is in the second object's message path, a runaway recursion will result. For example, if the following handler is in a stack script, and you get the "myCustomProperty" of a card in that stack, runaway recursion will result:

```
getProp myCustomProperty
  put the myCustomProperty of the target into myVariable
  -- Because the target is the card, and this handler is in
  -- the stack, the above statement sends another getProp call
  -- to the card.
end myCustomProperty
```

To avoid this problem, set the `lockMessages` property to true before checking the custom property.

You can include as many `getProp` handlers in a script as you need. The property that a `getProp` handler controls is determined by the `propertyName` parameter in the first line of the handler. (If a script contains two `getProp` handlers for the same property, the first one is used.)

Note: You cannot use a `getProp` handler to intercept a call for the value of a built-in property. The `getProp` control structure can be used only for custom properties.

A `getProp` handler can be used to implement virtual properties for an object. A virtual property does not exist in the list of the object's custom properties. Instead, when a statement calls for the property's value, the `getProp` handler computes that value.

For example, the following `getProp` handler implements a virtual property of a field that contains a list of numbers:

```
getProp columnAverage
  repeat for each line thisNumber of me
    add thisNumber to fieldTotal
  end repeat
  return fieldTotal/the number of lines of me
end columnAverage
```

The "columnAverage" property of the field does not exist in the list of the field's custom properties. Instead, it is evaluated when a statement requests it:

put the columnAverage of field "Numbers" into field "Average"

getResource

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

copyResource function, deleteResource function, getResources function, resfile keyword, setResource function, sysError function

Summary

Returns the data in a Mac OS resource.

Syntax

getResource(filePath,resourceType,{resourceName | resourceID})

Example Code

```
getResource("/Hard Drive/Stuff","DLOG",128)
getResource(it,"BNDL","Application")
```

Comments

Use the getResource function to get the contents of a resource.

Parameters:

The filePath is the location and name of the file that holds the resource you want. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

The resourceType is the 4-character type of the resource you want.

The resourceName is a string or an expression that evaluates to a string.

The resourceID is an integer or an expression that evaluates to an integer.

Value:

The getResource function returns the text or binary data in the specified resource.

If the specified resource does not exist, the getResource function returns empty.

Comments:

If the filePath does not exist, the result is set to "can't find file". If the filePath exists, but the file has no resource fork, the result is set to "can't open resource fork". If the filePath contains a resource fork but does not contain the specified resource, the result is set to "can't find specified resource".

getResources

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

copyResource function, deleteResource function, getResource function, resfile keyword, sysError function

Summary

Returns a list of the resources in a Mac OS file.

Syntax

getResources(filePath[,resourceType])

Example Code

```
getResources("Project Resources")
getResources(it,"STR#")
```

Comments

Use the getResources function to find out whether a resource already exists before using it or copying it.

Parameters:

The filePath is the location and name of the file whose resource fork you want to list. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

The resourceType is the 4-character type of the resources you want to list. If you don't specify a resourceType, the getResources function lists all the resources of all resource types.

Value:

The getResources function returns a list of resources, one per line. Each line consists of four items:

- the 4-character resource type

- the resource ID

- the resource name

- the resource size in bytes

- one or more resource flag characters. The possible resource flags are as follows:

 - S System heap

 - U Purgeable

L	Locked
P	Protected
R	Preload
C	Compressed resource

If a flag is set to true, its character is included in the last item. If the flag is set to false, its character is not included. If none of these flags is set for a resource, the last item of that resource's line is empty.

If the file does not contain any resources, the getResources function returns empty.

Comments:

If the filePath does not exist, the result is set to "can't find file". If the filePath exists, but the file has no resource fork, the result is set to "can't open resource fork".

If the file has a resource fork but no resources, the result is empty.

Changes to Transcript:

The order of items returned by the getResources function was changed in version 1.1. In previous versions, the getResources function returned these items for each resource:

- the resource name
- the resource ID
- the 4-character resource type
- the resource size in bytes

The resource flags item of the return value was introduced in version 1.1. In previous versions, the resource flags were not available.

global

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command, delete variable command, globalNames function, local command, About containers, variables, and sources of value

Summary

Declares a global variable.

Syntax

global variableNameList

Example Code

```
global currentDate  
global thisThing, thatThing, theOtherThing
```

Comments

Use the global command to define a variable that can be used in any handler, and which retains its value between handlers.

Parameters:

The variableNameList is a list of global variables separated by commas.

Comments:

You can place the global command either in a handler, or in a script but outside any handler in the script:

ï If you declare the global in a handler, the global declaration must appear in each handler in which you use the global. If you declare a global in one handler, and try to use it in another without first declaring it in that handler, the second handler treats it as a local variable, and it does not retain its value between handlers.

The global command can appear anywhere in a handler, as long as it's before the first statement in which the global variable is used. However, to make them easier to find, all global declarations are usually placed at the beginning of a handler:

```
on mouseUp
  global userName
  ask "Please enter your favorite color," && userName
  set the backdrop to it
end mouseUp
```

ï If you declare a global command in a script, but outside any handlers in the script, the global can be used by any handler that comes after the global declaration in that script. You don't need to declare such a global again in the handler itself.

Such global commands are usually placed at the beginning of the script, before any handlers, to make them easy to find:

```
global userName

on mouseUp
  ask "Please enter your favorite color," && userName
  set the backdrop to it
end mouseUp
```

globalLoc

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clickLoc function, foundLoc function, localLoc function, mouseLoc function, screenMouseLoc property, screenRect function, selectedLoc function

Summary

Returns the equivalent, in global coordinates, of a point given in local coordinates.

Syntax

the globalLoc of point
globalLoc(point)

Example Code

```
globalLoc("22,173")  
put globalLoc(the mouseLoc) into screenLoc
```

Comments

Use the globalLoc function to translate between screen coordinates and window coordinates.

Parameters:

The point is any expression that evaluates to a point—a vertical and horizontal distance from the top left of the current stack, separated by a comma.

Value:

The globalLoc function returns two integers separated by a comma.

Comments:

In window coordinates, the point 0,0 is at the top left of the stack window. In screen coordinates, the point 0,0 is at the top left of the screen.

The point returned by the globalLoc function is relative to the top left of the screen. If the system has more than one monitor, the globalLoc function returns a point relative to the top left of the main screen.

The first item of the return value is the horizontal distance in pixels from the left edge of the screen to the location given by point. The second item of the return value is the vertical distance from the top edge of the screen to the location given by point.

globalNames

function

Synonyms

globals

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete variable command, global command, localNames function, params function, variableNames function

Summary

Returns a list of declared global variables.

Syntax

the globalNames

globalNames()

Example Code

```
the globalNames
```

```
if "myGlob" is not among the items of the globalNames then global myGlob
```

Comments

Use the globalNames function to determine which global variables are available, or to make sure a global variable name has not already been used before declaring it.

Value:

The globalNames function returns a list of all global variables, separated by commas.

Comments:

Global variables are created with the global command. They can be deleted with the delete variable command. All global variables that have been declared since the application was started up are included in the globalNames function, except any that have been explicitly deleted.

go
command

Synonyms
open

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close command, drawer command, find command, ID property, lockRecent property, mark property, modal command, modeless command, name property, number property, palette command, recentNames property, topLevel command, How to bring a window to the front, How to go to another card, How to use a stack that's on a web server, How to open a stack, How to respond when a stack opens, Why can't I open a downloaded stack?, Why does Revolution ask to purge a stack?, Why is there already a stack with the same name?, Recipe for a Cards menu, Recipe for a Window menu, File menu > Open Stack..., View menu > Go First, View menu > Go Prev, View menu > Go Next, View menu > Go Last, Shortcut to go to the previous card, Shortcut to go to the next card

Summary

Navigates to another card or stack.

Syntax

```
go [invisible] [to] card [of stack] [as mode|in [a] new window|in window]
go [invisible] [to] {first | prev[ious]| next | last | any} [marked] [card]
go [invisible] [to] {recent | start | finish | home} card
go [invisible] [to] {forward | forth | back[ward]} [number]} [card[s]]
```

Example Code

```
go to card "Hemingway" -- in the defaultStack
go to stack "Controls" as palette
go to next marked card
go back 7 cards
go invisible stack "Preferences"
go stack URL "http://www.example.org/data/mystack.rev" in a new window
```

Comments

Use the go command to move to another card in the current stack, to open a stack and go to a card within it, or to move backward and forward among recently visited cards.

Parameters:

The card is any card reference. Cards can be described by their name, number, or ID properties.

The stack is any stack reference, or any file path or http URL reference that resolves to a stack file. If you specify a file path or http URL, the command opens the main stack of the specified stack file.

The mode is one of the following:

- topLevel: editable window
- palette: palette
- modal: modal dialog box
- modeless: modeless dialog box
- sheet: sheet dialog box (appears in defaultStack)
- drawer: drawer (appears in defaultStack, centered at left side if there's room)

The window is the name or windowID property of any open stack. If a window is specified, the stack opens in that window, replacing the stack that was previously displayed in that window.

The number is the number of cards to move within the recent cards list.

Comments:

If the stack is open, is closed but loaded into memory, or is listed in the current stack's stackFiles property, you can specify it simply by name:

```
go stack "My Stack"
```

Otherwise, you must include the stack's file path.

When going to a previously-unopened stack, if you don't specify a card, the go command displays the first card of the stack. If the stack is already open, the current card of the stack appears and the stack window is brought to the front.

If the lockScreen property is set to true, the go command does not bring an already-open stack to the front until the lockScreen is set to false. (Remember that the lockScreen is automatically set to false when all pending handlers finish executing.)

If you specify a mode, the stack opens in the specified mode. If you don't specify a mode, the stack opens in whatever mode is specified by the stack's style property.

When going to a stack, you can specify a mode or a window for the stack to appear in, but not both.

Important! The style of the stack, if it is anything other than "topLevel", overrides any mode you specify in a go command. For example, if you open a stack using the statement go stack "Some Stack" as modeless, and the style of "Some Stack" is set to "palette", it opens as a palette rather than a modeless dialog box, ignoring the mode you specified.

The go...as sheet form can be used only on OS X systems. If you use this form on Mac OS, Unix, or Windows, the stack is displayed as a modal dialog box instead. If you don't specify a mode, the stack is opened with the mode specified by its style property.

If you specify a URL, the stack is downloaded from that URL and displayed. The stack must be in stack file format (that is, not compressed or archived). Stacks opened in this way are treated as unsaved stacks; the long name of such a stack is the same as its abbreviated name, until the stack is saved on a

local disk. The downloaded stack is a copy: changes you make to the stack are not automatically made on the server the stack came from. To change the stack on the server, you must save the stack file locally and re-upload it.

You can go to the first, previous, next, or last card of the current stack. The form go any card goes to a random card in the current stack. If you include the marked parameter, the go command is restricted to cards whose mark property is set to true.

Each card the user visits while the lockRecent property is false is placed in the recent cards list. You can use the go command to move among the previously-visited cards:

- The go recent card form goes to the most recently visited card.
- The go start and go finish forms go to the first or last card in the recent cards list.
- The go home form goes to the first card in the application's main stack. (This form is included mainly for HyperCard compatibility; the statement go home does not do anything useful in the development environment.)
- The go back number form backs up number cards. The go forward number form moves forward in the recent cards list. "forward" and "forth" are synonyms.

If you use the go invisible form, the window or card change does not show on the screen. (When going to a stack, this form sets the stack's visible property to false.) Use this form of the go command to open a stack without displaying it on screen. To display the stack later, use the show command or set its visible property to true.

Any visual effects that have been queued with the visual effect command are displayed when the go command is executed (unless the screen is locked).

If the stack or card you specify doesn't exist, an error message is returned by the result function.

Tip: To test whether a stack or card exists before trying to go to it, use the there is a operator:

```
if there is a card "Index" then go card "Index"  
if there is a stack myPath then go card 2 of stack myPath
```

Changes to Transcript:

The go...as sheet form was introduced in version 2.0. Previous versions did not support sheet dialogs.

grab

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

click command, drag command, mouseDown message, mouseLoc function, mouseMove message, move command

Summary

Causes an object to follow the movements of the mouse.

Syntax

grab object

Example Code

```
grab graphic "Turtle"  
grab the mouseControl  
grab me
```

Comments

Use the grab command within a mouseDown handler to drag an object around the stack window without selecting it.

Parameters:

The object is any control on the current card.

Comments:

You can only grab a control when the mouse pointer is within the control's rectangle at the time the mouse is clicked. If the mouse pointer is outside the control when the grab command is executed, nothing happens.

When the user releases the mouse button, the grab command exits, the handler it's in resumes, and the control is dropped wherever it was last dragged.

graphic

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

button keyword, choose command, field keyword, image keyword, player keyword, scrollbar keyword, style property, templateGraphic keyword, tool function

Summary

Designates the Graphic tool, which is used to create new graphics.

Syntax

Example Code

```
choose graphic tool
```

Comments

Use the graphic keyword to create graphics.

Comments:

When using the Graphic tool, the cursor is a crosshairs. This tool is used for creating graphics by clicking at one corner of the graphic and dragging to the opposite corner.

You can set the style of the templateGraphic to the graphic type you want, then draw the graphic with the Graphic tool. The graphic icons on the Tools palette work this way: each one sets the style of the templateGraphic, then chooses the Graphic tool so you can create the graphic.

graphic object

Synonyms
grc

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

templateGraphic keyword, About object types and object references, Object menu > New Control > Rectangle Graphic, Object menu > New Control > Oval Graphic, Object menu > New Control > Curve Graphic, Object menu > New Control > Round Rect Graphic, Object menu > New Control > Polygon Graphic, Object menu > New Control > Line Graphic, Object menu > New Control > Regular Polygon Graphic

Summary

A control that is a resizeable geometric shape.

Syntax

Example Code

```
set the backgroundPattern of graphic "High Energy" to 3012  
move graphic "Arrow" to 45,104 in 3 seconds
```

Comments

Use the graphic object type to create a geometric shape, a straight or broken line, an arrow, or other shape.

Comments:

Graphics can be circles, ovals, arcs, squares, rectangles, regular polygons with any number of sides, irregular polygons (closed or open), lines (jagged or straight), or curves (smooth or broken). You specify a graphic's basic shape by setting its style property, and details of different basic shapes with the points, arcAngle, startAngle, angle, and polySides properties.

Unlike an image, a graphic can be resized without losing detail or becoming "jagged".

A graphic is contained in a card, group, or background. Graphics cannot contain other objects.

The graphic object has a number of properties and messages associated with it. To see a list of messages that can be sent to a graphic as a result of user actions or internal Revolution events, open the

"Transcript Language Dictionary" page of the main Documentation window, and choose "Graphic Messages" from the Show menu at the top. To see a list of all the properties a graphic can have, choose "Graphic Properties" from the Show menu.

gray
keyword

Synonyms
grey

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

black keyword, card keyword, hide command, inverse keyword, show command, visual effect command, white keyword

Summary

Used with the visual effect command, to show a blank gray screen at the end of the visual effect.

Syntax

Example Code

```
visual effect dissolve to gray
```

Comments

Use the gray keyword to transition to a blank gray background in a sequence of visual effects.

Comments:

Visual effects can be stacked in a sequence by using several visual effect commands in succession. If the last transition ends with showing the destination card, and all except the last one shows an intermediate image (such as a solid gray color), the effect is enhanced. You show a solid gray color at the end of a transition by using the gray keyword.

The gray background is actually a checkerboard pattern of alternating black and white pixels.

This example dissolves the image on the screen through an intermediate blank gray screen:

```
visual effect dissolve to gray -- from card to solid gray  
visual effect dissolve to card -- from gray to final card  
go card "Destination"
```

gRevAppIcon

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

answer command, ask command, ask password command, gRevSmallAppIcon keyword, ID property

Summary

A special global variable that specifies the application icon to be used on OS X systems with the ask, ask password, and answer commands.

Syntax

Example Code

```
global gRevAppIcon  
put the short ID of image "Icon" into gRevAppIcon
```

Comments

According to Aqua user-interface standards, the application's icon should appear in modal dialog boxes to identify which application the dialog box belongs to. Use the gRevAppIcon keyword to specify which image to use for the icon.

Comments:

In the Revolution development environment, when running on an OS X system, the Revolution icon is displayed in the ask, ask password, and answer dialog boxes. In your own applications, however, the application's icon should appear instead.

To make the correct icon appear, create an image in one of your application's stacks. (The image does not need to be visible, and the stack window it's in does not need to be open. The stack file needs only to be loaded into memory for the application to be able to use the image.)

Note: The standard size for the application icon is 64x64 pixels.

Make a note of the image's ID. Then, in your application, put the ID number into the gRevAppIcon global variable. For example, if the image ID is 3445, you can place the following statements in your application's startup handler:

```
global gRevAppIcon  
put "3445" into gRevAppIcon
```

Thereafter, the ask, ask password, and answer commands will use the image you specified.

Note: If you specify an iconType in the answer, ask, or ask password command, the image specified by the gRevSmallAppIcon variable appears instead, along with the standard icon specified by the iconType.

gRevProfileReadOnly

keyword

Synonyms

Objects

Profile library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer command, ask command, ask password command, gRevSmallAppIcon keyword, ID property

Summary

A special global variable that specifies whether to save changes to the current profile when you switch profiles.

Syntax

Example Code

```
global gRevProfileReadOnly
if gRevProfileReadOnly is false then switchToLargeFonts
```

Comments

Set the gRevProfileReadOnly variable to false when configuring a profile, so that changes you make to an object's properties are saved in the current profile.

Comments:

Each object can have one or more profiles, which include settings for each property in the object's properties property. The gRevProfileReadOnly variable controls what happens when you set an object's profile, make changes to its properties, then switch to a different profile.

If this global variable is false, the profile is re-saved when you switch profiles, and any property changes you've made are saved in the profile. If it's true, the profile isn't saved when you switch to another profile.

The gRevProfileReadOnly global variable can also be changed in the Preferences dialog box:

1. Choose Edit menu Preferences.
2. Choose "Property Profiles" from the menu at the top.

3. Click "Don't save changes in profile".

gRevSmallAppIcon

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

answer command, ask command, ask password command, gRevAppIcon keyword, ID property

Summary

A special global variable that specifies the small application icon to be used on OS X systems with the ask, ask password, and answer commands.

Syntax

Example Code

```
global gRevAppSmallIcon  
put 8799 into gRevAppSmallIcon
```

Comments

According to Aqua user-interface standards, the application's icon should appear in modal dialog boxes to identify which application the dialog box belongs to. Use the gRevSmallAppIcon keyword to specify which image to use for the icon.

Comments:

In the Revolution development environment, when running on an OS X system, the Revolution icon is displayed in the ask, ask password, and answer dialog boxes. In your own applications, however, the application's icon should appear instead. (This icon is specified by the gRevAppIcon keyword.)

If you specify an iconType (information, question, error, or warning) with the answer, ask, or ask password command, the standard icon for that type of dialog box appears instead, "badged" with a small version of your application's icon. You specify that small icon in the gRevSmallAppIcon variable.

To make the correct icon appear, create an image in one of your application's stacks. (The image does not need to be visible, and the stack window it's in does not need to be open. The stack file needs only to be loaded into memory for the application to be able to use the image.)

Note: The standard size for the small application icon is 32x32 pixels.

Make a note of the image's ID. Then, in your application, put the ID number into the `gRevSmallAppIcon` global variable. For example, if the image ID is 3446, you can place the following statements in your application's startup handler:

```
global gRevSmallAppIcon  
put "3446" into gRevSmallAppIcon
```

Thereafter, the `answer`, `ask`, and `ask password` commands will use the image you specified.

grid

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

gridSize property, hGrid property, vGrid property, View menu > Grid

Summary

Specifies whether dragged objects are snapped to a grid.

Syntax

set the grid to {true | false}

Example Code

```
set the grid to true
```

Comments

Use the grid property to align objects easily.

Value:

The grid is true or false

By default, the grid property is set to true.

Comments:

If the grid property is set to true, any object the user moves or resizes is moved or resized to the nearest point on the grid. If the object is moved, its top and left are multiples of the gridSize. If the object is resized, its height and width are multiples of the gridSize.

If the grid is false, moved and resized objects are not constrained and are moved to exactly where the user puts them.

The setting of the grid property does not affect the move command, or setting properties such as the location or rectangle of the object, or nudging objects a single pixel with the arrow keys. Only mouse movements are affected.

gridSize

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

centered property, grid property, Edit menu > Preferences

Summary

Specifies the size of squares in the drag grid.

Syntax

set the gridSize to pixelSize

Example Code

```
set the gridSize to 10 -- objects are aligned to a 10-pixel mesh
```

Comments

Use the gridSize property to align objects in a coarse or fine grid.

Value:

The gridSize is a positive integer.

By default, the gridSize property is set to 4.

Comments:

If the grid property is set to true, any object the user moves or resizes is moved or resized to the nearest point on the grid. If the object is moved, its top and left are multiples of the gridSize. If the object is resized, its height and width are multiples of the gridSize.

If the grid is false, the setting of the gridSize property has no effect.

group command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

layer property, newGroup message, place command, remove command, selectedObject function, ungroup command, About groups and backgrounds, Object menu > Group Selected

Summary

Creates a new group and adds the selected objects to it.

Syntax

group [objectList]

Example Code

```
group
if the number of lines of the selectedObject > 1 then group
group field "Help" and image "Help Icon" and graphic "Outline"
```

Comments

Use the group command after selecting objects to make a group out of them.

Parameters:

The objectList is a list of object references, separated by the word "and".

Comments:

If you specify an objectList, the specified objects are placed in the new group. Otherwise, the selected object or objects are placed in the group.

As with any group on a card, you can use the place command to make the newly-created group appear on other cards.

Tip: To refer to the newly-created group, use the last keyword:

```
group button "Yes" and button "No"
set the name of last group to "Do It"
```

group object

Synonyms

grp, background, bkgnd, bg

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

templateGroup function, About groups and backgrounds, About object types and object references, Object menu > Group Selected, Object menu > Edit Group, Object menu > Remove Group, Object menu > Place Group

Summary

A control that contains other controls.

Syntax

Example Code

```
set the showBorder of last group to true  
create group "Options"
```

Comments

Use the group object type to hold sets of controls, radio-button clusters, menu bars, and controls to be displayed on more than one card.

Comments:

A group is a set of controls that has been made into a single control. You can select, move, resize, or copy the group, and all the controls in it come with the group. You can show a border around the group (using its border property), a label (using the showName property), or scrollbars.

Groups can contain any type of control (including other nested groups).

When referring to a group using the synonyms background, bkgnd, or bg, the reference is to one among the groups in a stack, rather than to one among the groups on a card. For example, the object reference background 1 indicates the first group in the current stack, not the lowest-layered group on the current card.

The group object has a number of properties and messages associated with it. To see a list of messages that can be sent to a group as a result of user actions or internal Revolution events, open the "Transcript

Language Dictionary" page of the main Documentation window, and choose "Group Messages" from the Show menu at the top. To see a list of all the properties a group can have, choose "Group Properties" from the Show menu.

groupIDs

property

Synonyms

Objects

card

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

backgroundIDs property, cardIDs property, editBackground property, groupNames property, ID property, place command, remove command, start editing command, About groups and backgrounds, Why isn't a group listed?

Summary

Reports the groups on a card.

Syntax

get the groupIDs of card

Example Code

```
repeat with x = 1 to the number of lines of the groupIDs of this card
```

Comments

Use the groupIDs property to find out which groups are placed on a card.

Value:

The groupIDs of a card reports a list of short ID properties of groups, one per line.

This property is read-only and cannot be set.

Comments:

If a group on the card contains groups, only the top-level groups are reported.

To find out which groups are in a stack, use the backgroundIDs property.

Changes to Transcript:

In versions before 1.1, groupIDs and backgroundIDs were synonyms and could be used interchangeably.

groupNames

property

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundNames property, cardNames property, editBackground property, groupIDs property, place command, remove command, start editing command, About groups and backgrounds, Why isn't a group listed?

Summary

Reports the groups on a card.

Syntax

get the groupNames of card

Example Code

```
put the groupNames of card 1 into groupsToDelete
if x is among the lines of the groupNames of card y then next repeat
```

Comments

Use the groupNames property to find out which groups are placed on a card.

Value:

The groupNames of a card reports a list of group names, one per line.

This property is read-only and cannot be set.

Comments:

If a group on the card contains groups, only the top-level groups are reported.

To find out which groups are in a stack, use the backgroundNames property.

Changes to Transcript:

In versions before 1.1, groupNames and backgroundNames were synonyms and could be used interchangeably.

hand

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command, cursor property, eight constant, one constant

Summary

Equivalent to the number 8.

Syntax

Example Code

```
set the cursor to hand
```

Comments

Use the hand constant to set the cursor to a pointing hand shape.

Comments:

The following two statements are equivalent:

```
set the cursor to hand
```

```
set the cursor to 8
```

However, the first is easier to read and understand in Transcript code.

Important! If you use the hand cursor or other standard cursors in your application, you must include it when you create your standalone. Make sure the "Cursors" option on the Inclusions tab in Step 3 of the Distribution Builder is checked.

hasMemory

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

alwaysBuffer property, delete stack command, delete variable command, destroyStack property, diskSpace function, heapSpace function, revUnloadSpeech command, Memory and Limits Reference, How to estimate how much memory your application will need, Why am I running out of memory?

Summary

Returns true if the specified amount of memory is available, false otherwise.

Syntax

the hasMemory of bytes

hasMemory(bytes)

Example Code

```
hasMemory(2*1024*1024) -- returns true if 2M available  
if hasMemory(500*1024) then open stack "Explore Multimedia"
```

Comments

Use the hasMemory function to check whether there's enough memory available for an action (such as displaying a large graphic) before you do it.

Parameters:

The bytes is the number of bytes you require. If the amount of available memory is greater than or equal to the bytes, the function returns true.

Comments:

This function is only partially implemented, and may not return useful values on some platforms. It is included in Transcript for compatibility with imported SuperCard projects.

Cross-platform note: On Mac OS systems, the value returned depends on the amount of free memory in the application heap and does not take available temporary memory into account. This means that even if the hasMemory function returns false, there may be enough system memory available for the task, since Revolution uses temporary memory when necessary.

HCAddressing

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundBehavior property, dynamicPaths property, HCStack property, layer property, number property, About Revolution for HyperCard developers, How to import a HyperCard stack, Why does importing a HyperCard stack cause an error?

Summary

Determines whether grouped fields and card buttons are assumed if the field or button's domain is not specified.

Syntax

set the HCAddressing of stack to {true | false}

Example Code

```
set the HCAddressing of stack nextImport to true
```

Comments

Use the HCAddressing property for compatibility with HyperCard.

Value:

The HCAddressing of a stack is true or false.

By default, the HCAddressing of stacks you create in Revolution is set to false.

Comments:

When you open a HyperCard stack and convert it to a Revolution stack, the new stack's HCAddressing property is set to true.

If the HCAddressing property is set to true, expressions in the stack's scripts that refer to fields without specifying card or background are assumed to refer to grouped controls, and expressions that refer to other control types are assumed to refer to card controls. For example, the number of fields reports the number of grouped fields, while button 5 refers to the fifth card button. Also, the values reported by the long or abbreviated name and ID properties of controls include "card" or "background" as the first word, depending on whether the control is part of a group.

If the HCAddressing property is false, expressions that refer to fields or buttons without specifying card or background are assumed to refer to all fields or buttons. For example, the number of fields reports the total number of card and background fields, and button 5 refers to the fifth button.

If a stack's HCAddressing property is true, the style property of a button whose style is "menu" reports "popup" instead. This is because HyperCard uses a style setting of "popup" to designate popup menus.

HCImportStat

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

go command, HCStack property, About Revolution for HyperCard developers, How to import a HyperCard stack, Why does importing a HyperCard stack cause an error?

Summary

Reports information about stacks that are being imported from HyperCard.

Syntax

get the HCImportStat

Example Code

```
put the HCImportStat into field "Progress"
```

Comments

Use the HCImportStat property to check the status of a HyperCard stack that's being imported.

Value:

The HCImportStat property reports an expression of the form "Loading stack stackNameÖ" while a HyperCard stack is being imported.

Comments:

To import a HyperCard stack, use the go command to open the stack. Revolution automatically converts it to a Revolution stack.

If no stack has been imported during the current session, the HCImportStat property reports empty.

You can set this property back to empty after a stack has been successfully imported.

HCStack

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

HCAAddressing property, HCImportStat property, About Revolution for HyperCard developers, How to import a HyperCard stack, Why does importing a HyperCard stack cause an error?

Summary

Reports whether a stack was originally imported from HyperCard.

Syntax

get the HCStack of stack

Example Code

```
set the HCAAddressing of this stack to the HCStack of this stack
```

Comments

Use the HCStack property to find out whether you need to adjust properties for compatibility with scripts imported from HyperCard.

Value:

The HCStack of a stack is true or false.

This property is read-only and cannot be set.

Comments:

If the specified stack was originally a HyperCard stack that was imported into Revolution, its HCStack property is true.

If the specified stack was never a HyperCard stack, its HCStack property is false.

heapSpace

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

alwaysBuffer property, delete stack command, delete variable command, destroyStack property, diskSpace function, hasMemory function, Memory and Limits Reference, How to estimate how much memory your application will need, Why is Revolution taking more memory than its allocation?

Summary

Returns the number of free bytes in the application heap on Mac OS systems.

Syntax

the heapSpace

heapSpace()

Example Code

```
the heapSpace  
if the heapSpace < 1024^2 then suggestLowMemory
```

Comments

Use the heapSpace function to determine how much free memory remains.

Value:

The heapSpace function returns a positive integer.

Comments:

This function exists to aid compatibility with imported HyperCard stacks. It may not return the exact amount of free memory. It also does not take available temporary memory into account.

On Unix and Windows systems, this function does not return a meaningful value.

height property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

bottom property, editMenus property, formattedHeight property, lockLocation property, maxHeight property, minHeight property, rectangle property, revChangeWindowSize command, screenRect function, top property, width property, How to determine the dimensions of an image, How to prevent the user from resizing a window, How to put different-sized cards in the same stack, Recipe for adding a scrollbar to a field when the contents overflow it, Object menu > Align Selected Controls > Make Heights Equal, Shortcut to equalize heights of selected controls

Summary

Specifies the distance from an object's top edge to its bottom edge.

Syntax

set the height of object to numberOfPixels

Example Code

```
put the height of button 1 + the height of button 2 into requiredHeight
set the height of field thisField to the textHeight of thisField
```

Comments

Use the height property to determine how much vertical space an object needs, or to make it taller or shorter.

Value:

The height of an object is a non-negative integer.

Comments:

If an object's lockLocation property is false, when you change its height, it shrinks or grows from the center. The object's top and bottom edges both shift, while the object's location property stays the same. If the object's lockLocation property is true, it shrinks or grows from the top left corner: the object's top edge stays in the same place, and the bottom edge moves.

If you reduce the height of a stack, some objects may end up outside the stack window. These objects are not shown; however, they are still there, and will be displayed if you make the window tall enough.

Cross-platform note: On Mac OS and OS X systems, the menu bar normally appears at the top of the screen, rather than inside the stack window. If a stack contains a menu bar and the stack's `editMenus` property is set to `false`, the stack window is automatically resized so that the menu bar group is not visible in the window. In this case, the height of the stack is the current height of the stack window., but the height of the card is the height of the total content area of the stack (including the hidden menu bar group). This is equal to the height of the stack plus its `vScroll`.

You can set the height of a card, but doing so has no effect and doesn't change the card's height property.

help

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command, cursor property

Summary

Equivalent to the number 15.

Syntax

Example Code

```
set the cursor to help
```

Comments

Use the help constant to set the cursor to a question mark.

Comments:

The following two statements are equivalent:

```
set the cursor to help
```

```
set the cursor to 15
```

However, the first is easier to read and understand in Transcript code.

Important! If you use the help cursor or other standard cursors in your application, you must include it when you create your standalone. Make sure the "Cursors" option on the Inclusions tab in Step 3 of the Distribution Builder is checked.

help

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

functionKey message, keyDown message, Help menu, Shortcut to open Revolution documentation

Summary

Sent when the user presses the Help or F1 key.

Syntax

help

Example Code

```
on help -- bring up your own help, if running as standalone
  if the environment is "development"
    then pass help
  else go stack "My Help Tips"
end help
```

Comments

Handle the help message in order to substitute custom help for the Revolution help system, or to provide help in a standalone application.

Comments:

The message is sent to the active (focused) control, or to the current card if no control is focused.

hGrid

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

borderColor property, fixedLineHeight property, showLines property, textHeight property, vGrid property

Summary

Specifies whether lines are shown below each text line in a field.

Syntax

set the hGrid of field to {true | false}

Example Code

```
set the hGrid of field 3 to true
```

Comments

Use the hGrid property to control the appearance of a field.

Value:

The hGrid of a field is true or false.

By default, the hGrid property of newly created fields is set to false.

Comments:

If a field's hGrid property is true, a horizontal line is drawn across the field, every textHeight pixels. The line falls below the lowest part of each character, rather than being drawn at the text baseline, as the lines drawn by the showLines property are.

The lines shown by the hGrid and vGrid are drawn in the borderColor (or borderPattern).

If the field's fixedLineHeight property is set to false, the hGrid property has no effect.

Changes to Transcript:

The use of the borderColor to draw grid lines was introduced in version 2.0. In previous versions, the grid lines were drawn in the hiliteColor.

hide

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hide groups command, hide menubar command, show command, showInvisibles property, visible property, visual effect command, How to hide the Toolbar, How to hide other applications' windows, How to peek at invisible objects in a stack, Why don't visual effects work?, Recipe for a "roll credits" effect, Development menu > Suspend Development Tools

Summary

Makes an object invisible.

Syntax

hide object [with visual [effect] effectName [speed] [to finalImage]]

Example Code

```
hide button 14
hide field "New" with visual effect dissolve
hide this stack with visual dissolve fast to black
```

Comments

Use the hide command to hide objects temporarily or permanently.

Parameters:

The object is any open stack, or any control in an open stack.

The effectName, speed, and finalImage are described in the visual effect command. These three parameters operate the same way as they do in the visual effect command.

Comments:

Hiding an object sets its visible property to false. The object itself is still in the stack and can be restored with the show command. You can also set the showInvisibles property to true to override the visible property and show all objects.

Hiding a stack hides its window, but does not close the stack. If the stack is being displayed as a drawer, hiding it slides it back into its parent stack.

You can hide a card without causing a script error, but doing so has no effect.

Hiding a group hides all the controls in the group.

Changes to Transcript:

The ability to use QuickTime special effects was introduced in version 1.1. In previous versions, only the built-in visual effects listed under the visual effect command were available.

hide groups

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

hide command, hide menubar command, link keyword, underlineLinks property, show groups command

Summary

Removes the underline from grouped text.

Syntax

hide groups

Example Code

```
hide groups  
if the lookAndFeel is "Motif" then hide groups
```

Comments

Use the hide groups command to remove the underline from grouped text.

Comments:

Text with its textStyle set to "link" is treated specially by the clickText, clickChunk, mouseText, and mouseChunk functions: a style run of grouped text is treated as a single word. This makes grouped text handy to use for hypertext or "clickable text" features.

The hide groups command sets the global underlineLinks property to false. It does not affect the stack underlineLinks property, so if a stack's underlineLinks is true, hide groups does not remove the underlining in that stack.

hide menubar

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

hide command, hide groups command, show menubar command

Summary

Hides the menu bar on Mac OS systems.

Syntax

hide menubar

Example Code

```
hide menubar  
if the commandKey is down then hide menubar
```

Comments

Use the hide menubar command to use the full screen. For example, in a kiosk or multimedia application, the menu bar is a distraction for users and in this case you may prefer to hide it.

Comments:

The menu bar can be hidden only on Mac OS and OS X systems. On OS X systems, the hide menubar command also hides the Dock. This command has no effect on Windows or Unix systems.

Hiding the menu bar in Revolution does not affect its appearance in other applications; if the user switches to another application, the menu bar becomes visible. If the user switches back to Revolution, the menu bar is hidden again.

hide taskbar

command

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

hide command, hide menubar command, show taskbar command

Summary

Hides the task bar on Windows systems.

Syntax

hide taskbar

Example Code

```
hide taskbar  
if the hilite of button "Kiosk Mode" then hide taskbar
```

Comments

Use the hide taskbar command to use the full screen. For example, in a kiosk or multimedia application, the task bar is a distraction for users and in this case you may prefer to hide it.

Comments:

The task bar can be hidden only on Windows systems. This command has no effect on Mac OS, OS X, or Unix systems.

hideConsoleWindows

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

launch command, open process command, shell function, shellCommand property, How to hide other applications' windows

Summary

Hides the main window of applications run with the open process command or shell function on Windows systems.

Syntax

set the hideConsoleWindows to {true | false}

Example Code

```
set the hideConsoleWindows to true
```

Comments

Use the hideConsoleWindows property to run command line applications without the user seeing them.

Value:

The hideConsoleWindows is true or false.

By default, the hideConsoleWindows property is set to false.

Comments:

On Windows systems, when you run a command line program with the launch or open process command, a console window appears. If the hideConsoleWindows property is true, the window is not shown. (You can still use the write to process and read from process commands to send data to the program and get data from it.)

The setting of this property affects Windows applications with a GUI as well as command line programs.

Important! If you run a program that requires user interaction, make sure the `hideConsoleWindows` property is set to `false` beforehand. Otherwise, the program will not be displayed and the user will not be able to see it.

The setting of this property has no effect on Mac OS or Unix systems.

hidePalettes

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

activatePalettes property, palette command, style property

Summary

Determines whether palette windows are hidden while Revolution is in the background.

Syntax

set the hidePalettes to {true | false}

Example Code

```
set the hidePalettes to false
```

Comments

Use the hidePalettes property to change the behavior of palettes.

Value:

The hidePalettes is true or false.

By default, the hidePalettes property is set to true on Mac OS and OS X systems, false on Unix or Windows systems.

Comments:

If the hidePalettes is true, palette windows—including stacks opened as palettes—are hidden when another application comes to the front, and are shown when Revolution is brought back to the front.

If the hidePalettes is false, palette windows are always shown, regardless of which application is active.

hilite

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

autoHilite property, enable command, hilite property, select command, unhilite command, How to change the highlight state of a checkbox or button, How to change the selected button in a radio button cluster

Summary

Highlights a button.

Syntax

hilite button

Example Code

```
hilite button "Overlay"  
hilite button nextButtonToShow  
hilite button ID (the buttonRecord of this card)
```

Comments

Use the hilite command to draw attention to a button, or to emulate the action of a button when it's being pressed, or to turn on a checkbox or radio button.

Parameters:

The button is any button reference.

Comments:

The hilite command sets the button's hilite property to true.

hilite

property

Synonyms

hilited, highlight, highlighted, highlite, highlited, hilight, highlighted

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

armed property, autoHilite property, hilite command, hiliteBorder property, hiliteColor property, hilitedButton property, hiliteFill property, hilitePattern property, mouseDown message, sharedHilite property, threeD property, unhilite command, How to change the highlight state of a checkbox or button, How to change the selected button in a radio button cluster, How to detect the highlight state of a control, How to give a checkbox a different state on each card

Summary

Determines whether a button is highlighted.

Syntax

set the hilite of button to {true | false}

Example Code

```
set the hilite of button "Option 1" to true
set the hilite of me to not the hilite of me
```

Comments

Use the hilite property to provide visual feedback when a button is pressed, or to reflect the state of a setting.

Value:

The hilite of a button is true or false.

By default, the hilite property of a newly created button is set to false.

Comments:

The effect of the hilite property on a button depends on the style, threeD, and hiliteBorder of the button.

If the button's hiliteBorder and threeD properties are true, the topColor and bottomColor (or topPattern and bottomPattern) are reversed when the button is highlighted.

If a radio button's or checkbox's hilite is true, the radio button or checkbox option is selected. If the hilite is false, the option is not selected.

For buttons of any other style, if the hilite is true, the button's background is drawn in the hiliteColor (or the hilitePattern, if the button's hiliteFill property is true).

hiliteBorder

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

armed property, bottomColor property, bottomPattern property, hilite property, lookAndFeel property, threeD property, topColor property, topPattern property

Summary

Specifies whether a three-D button appears to be pushed in when it is highlighted.

Syntax

set the hiliteBorder of button to {true | false}

Example Code

```
set the hiliteBorder of last button to true
```

Comments

Use the hiliteBorder property to affect the appearance of a three-D button when it is clicked.

Value:

The hiliteBorder of a button is true or false.

Comments:

If a button's hiliteBorder property is true, the button's highlight action is affected, depending on the style of the button. For radio buttons and checkboxes, a border is drawn around the entire button if its hilite is true. For other button styles, the button's effective topColor is used for the bottom and right edges, and the bottomColor is used for the top and left edges (reversing the normal effect), when the button is highlighted.

If the lookAndFeel property is "Macintosh", the hiliteBorder property has no effect on button styles other than radio buttons and checkboxes.

If the button's threeD property is false, the hiliteBorder property has no effect.

hiliteColor

property

Synonyms

markerColor, thirdColor

Objects

any object, global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

accentColor property, autoHilite property, backgroundColor property, borderColor property, bottomColor property, colorNames function, colors property, focusColor property, foregroundColor property, hilite property, hiliteFill property, hilitePattern property, owner property, shadowColor property, topColor property, About colors and color references, Color Names Reference, Why don't buttons respect my color settings?, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies the color of the background when an object, or text in an object, is highlighted.

Syntax

set the hiliteColor [of object] to {empty | colorName | RGBColor}

Example Code

```
set the hiliteColor of field "Comments" to "red"
set the hiliteColor of button ID 3 to "#336633"
set the hiliteColor to 255,255,0
```

Comments

Use the hiliteColor property to specify the background color of the selected text in a field, or the highlighted color of a clicked object.

Value:

The hiliteColor of an object is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the hiliteColor for all objects is empty.

By default, the global hiliteColor property is set to the system highlight color.

Comments:

Setting the hiliteColor of an object to empty allows the hiliteColor of the object's owner to show through. Use the effective keyword to find out what color is used for the object, even if its own hiliteColor is empty.

The setting of the hiliteColor property has different effects, depending on the object type:

- ĩ The hiliteColor of a stack, card, or group determines the hiliteColor of any object in the stack, card, or group that does not have its own hiliteColor.
- ĩ The hiliteColor of a button is used for the background of the button when it is highlighted. If the button is a menu, the hiliteColor is used to highlight the button, but not the active menu choice. The hiliteColor has no effect if the button is a tabbed button. The hiliteColor has no effect until the button is highlighted.
- Cross-platform note: If the lookAndFeel is set to "Appearance Manager", standard and rectangle buttons are drawn by the operating system if the backgroundColor and backgroundPattern of the button and all of its owners is empty. In this case, the button's hiliteColor has no effect. Otherwise, the button is drawn by Revolution. If the lookAndFeel is "Appearance Manager", button menus whose menuMode is set to "option" are always drawn by the operating system, and the setting of the button's hiliteColor does not affect them.
- ĩ The hiliteColor of a field determines the background color of text selections in the field. If the hiliteColor of the field and all its owners is empty, the global hiliteColor property is used.
- ĩ The hiliteColor of a scrollbar fills the arrow boxes at the ends of the scrollbar when the arrows are clicked. The hiliteColor has no effect until the arrows are clicked.
- ĩ The hiliteColor of a graphic outlines its marker shapes. (The hiliteColor has no effect unless the graphic's style is "curve" or "polygon" and its markerDrawn property is true.)
- ĩ The hiliteColor of a player or EPS object has no effect.
- ĩ The hiliteColor of an image is the third color in the image's color palette.

If the object's hilitePattern is set, the pattern is shown instead of the color specified by hiliteColor.

The global hiliteColor property specifies the background color of selected text in fields whose hiliteColor is empty, but does not affect other highlighted objects.

Changes to Transcript:

The ability of an object to inherit its hiliteColor from the object's owner was introduced in version 1.1. In previous versions, if an object's hiliteColor was empty, the setting of the global hiliteColor property was used instead.

The borderColor's effect on grid lines in fields was introduced in version 2.0. In previous versions, the color of the grid lines was determined by the field's hiliteColor property.

hilitedButton

property

Synonyms

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

autoHilite property, family property, hilite property, hilitedButtonID property, hilitedButtonName property, layer property, number property, radioButton property, How to change the selected button in a radio button cluster

Summary

Specifies which button in a group is currently highlighted.

Syntax

set the hilitedButton of group to buttonNumber

Example Code

```
set the hilitedButton of group "Options" to 2 -- 2nd button in group
set the hilitedButton of group ID 8 to zero -- unhilites all buttons
```

Comments

Use the hilitedButton property to change the current setting in a radio-button cluster.

Value:

The hilitedButton of a group is an integer between zero and the number of buttons in the group.

Comments:

The hilitedButton specifies which button in a group is currently highlighted. If the hilitedButton of a group is zero, none of the buttons in the group is currently highlighted. If the hilitedButton is 1, the button with the lowest layer in the group is currently highlighted; if the hilitedButton is 2, the button with the second-lowest layer is currently highlighted; and so forth.

Setting the hilitedButton of a group sets the hilite property of the rest of the buttons to false.

The hilitedButton property is most useful in radio-button clusters. In a radio-button cluster, only one button can be highlighted at once: clicking another button unhighlights the first one. The radioButton property of a group of buttons specifies whether they act as a radio-button cluster.

The `hilitedButton`, `hilitedButtonName`, and `hilitedButtonID` properties all refer in different ways to the same button. When any of them changes, all of them change.

hilitedButtonID

property

Synonyms

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

autoHilite property, family property, hilite property, hilitedButton property, hilitedButtonName property, ID property, radioBehavior property, How to change the selected button in a radio button cluster

Summary

Specifies which button in a group is currently highlighted.

Syntax

set the hilitedButtonID of group to buttonID

Example Code

```
set the hilitedButtonID of group "Options" to 342
set the hilitedButtonID of group 1 to the short ID of the mouseControl
```

Comments

Use the hilitedButtonID property to change the current setting in a radio-button cluster.

Value:

The hilitedButtonID of a group is zero or the short ID property of a button.

Comments:

The hilitedButtonID specifies which button in a group is currently highlighted. If the hilitedButtonID of a group is zero, none of the buttons in the group is currently highlighted.

Setting the hilitedButtonID of a group sets the hilite property of the rest of the buttons to false.

The hilitedButtonID property is most useful in radio-button clusters. In a radio-button cluster, only one button can be highlighted at once: clicking another button unhighlights the first one. The radioBehavior property of a group of buttons specifies whether they act as a radio-button cluster.

The hilitedButton, hilitedButtonName, and hilitedButtonID properties all refer in different ways to the same button. When any of them changes, all of them change.

hilitedButtonName

property

Synonyms

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

autoHilite property, family property, hilite property, hilitedButton property, hilitedButtonID property, name property, radioButton property, How to change the selected button in a radio button cluster

Summary

Specifies which button in a group is currently highlighted.

Syntax

set the hilitedButtonName of group to buttonName

Example Code

```
set the hilitedButtonName of group "Options" to "Help"
```

Comments

Use the hilitedButtonName property to change the current setting in a radio-button cluster.

Value:

The hilitedButtonName of a group is the short name property of a button.

Comments:

The hilitedButtonName specifies which button in a group is currently highlighted. If the hilitedButtonName of a group is empty, none of the buttons in the group is currently highlighted.

Setting the hilitedButtonName of a group sets the hilite property of the rest of the buttons to false.

The hilitedButtonName property is most useful in radio-button clusters. In a radio-button cluster, only one button can be highlighted at once: clicking another button unhighlights the first one. The radioButton property of a group of buttons specifies whether they act as a radio-button cluster.

The hilitedButton, hilitedButtonName, and hilitedButtonID properties all refer in different ways to the same button. When any of them changes, all of them change.

hilitedIcon

property

Synonyms

hiliteIcon

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hilite property, icon property, showIcon property, Why do icons disappear from a standalone application?

Summary

Specifies an image to display in a button when the button is highlighted.

Syntax

set the hilitedIcon of button to {imageID | imageName}

Example Code

```
set the hilitedIcon of button "Management" to 343
```

Comments

Use the hilitedIcon property to change a button's appearance when it is highlighted.

Value:

The hilitedIcon property is the ID or name of an image to use for an icon. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the hilitedIcon property of newly created buttons is set to zero.

Comments:

When the button is highlighted, the hilitedIcon is displayed within the button's rectangle.

Changes to Transcript:

The hilitedIcon keyword was introduced in version 1.1. In previous versions, the hiliteIcon synonym was used.

hilitedLine

property

Synonyms

hilitedLines

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickLine function, listBehavior property, multipleLines property, noncontiguousHilites property, selectedText function, toggleHilites property, How to select an entry in a list field, Why can't I leave all lines of a list field unselected?

Summary

Specifies the numbers of the selected lines in a list field.

Syntax

set the hilitedLine of field to listOfLines

Example Code

```
set the hilitedLine of field "Options List" to 1  
repeat with x = 1 to the number of items in the hilitedLines of me
```

Comments

Use the hilitedLine property to check which lines in a list field are selected, or to select a line.

Value:

The hilitedLine of a field is a list of one or more integers, separated by commas.

Comments:

Each item of the hilitedLine property is a line number. For example, if the first and fourth lines of the field are selected, the hilitedLine is 1,4. You can retrieve the text of the selected lines either by using the line numbers in a chunk expression or by using the selectedText function.

If the field's listBehavior property is false, this property has no effect.

hiliteFill

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hilite property, hiliteColor property, hilitePattern property

Summary

Specifies whether a button's background is filled with the hiliteColor and hilitePattern when the button is highlighted.

Syntax

set the hiliteFill of button to {true | false}

Example Code

```
set the hiliteFill of last button to false
```

Comments

Use the hiliteFill property to change the appearance a button has when it is highlighted.

Value:

The hiliteFill of a button is true or false.

By default, the hiliteFill of a newly created button is set to true.

Comments:

The hiliteFill property affects only the background of a button. If a button's hiliteFill property is false, its text changes color when highlighted, but its background color or pattern does not change.

hilitePattern

property

Synonyms

markerPattern, thirdPattern

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backgroundPattern property, borderPattern property, bottomPattern property, effective keyword, focusPattern property, foregroundPattern property, hiliteColor property, patterns property, shadowPattern property, topPattern property

Summary

Specifies the fill pattern used for the background when an object, or text in an object, is highlighted.

Syntax

set the hilitePattern of object to {patternNumber | imageID | empty}

Example Code

```
set the hilitePattern to 3859
```

Comments

Use the hilitePattern property to specify the background pattern used for the selected text in a field, or the highlighted pattern of a clicked object, or the pattern of graphic markers.

Value:

The hilitePattern of an object is a pattern specifier.

A patternNumber is a built-in pattern number between 1 and 164. (These patterns correspond to Revolution's built-in patterns 136 to 300.)

An imageID is the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the hilitePattern for all objects is empty.

Comments:

Pattern images can be color or black-and-white.

Cross-platform note: To be used as a pattern on Mac OS systems, an image must be 128x128 pixels or less, and both its height and width must be a power of 2. To be used on Windows and Unix systems, height and width must be divisible by 8. To be used as a fully cross-platform pattern, both an image's dimensions should be one of 8, 16, 32, 64, or 128.

The `hilitePattern` of controls is drawn starting at the control's upper right corner: if the control is moved, the pattern does not shift.

The setting of the `hilitePattern` property has different effects, depending on the object type:

- ï The `hilitePattern` of a stack, card, or group determines the `hilitePattern` of any object in the stack, card, or group that does not have its own `hilitePattern`.
- ï The `hilitePattern` of a button is used for the background of the button when it is highlighted. If the button is a menu, the `hilitePattern` is used to highlight the button, but not the active menu choice. The `hilitePattern` has no effect if the button is a tabbed button. The `hilitePattern` has no effect until the button is highlighted.

Cross-platform note: If the `lookAndFeel` is set to "Appearance Manager", standard and rectangle buttons are drawn by the operating system if the `backgroundColor` and `backgroundPattern` of the button and all of its owners is empty. In this case, the button's `hilitePattern` has no effect. Otherwise, the button is drawn by Revolution. If the `lookAndFeel` is "Appearance Manager", button menus whose `menuMode` is set to "option" are always drawn by the operating system, and the setting of the button's `hilitePattern` does not affect them.

- ï The `hilitePattern` of a field determines the background color of text selections in the field.
- ï The `hilitePattern` of a scrollbar fills the arrow boxes at the ends of the scrollbar when the arrows are clicked. The `hilitePattern` has no effect until the arrows are clicked.
- ï The `hilitePattern` of a graphic, image, player, or EPS object has no effect.

If the object's `hilitePattern` is set, the pattern is shown instead of the color specified by `hiliteColor`.

Note: Unlike the `hiliteColor` property, the `hilitePattern` is not a global property. Setting the `hilitePattern` of an object to empty causes the `hilitePattern` of the object's owner to be used for the object.

hilitePixel

property

Synonyms

thirdPixel

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundPixel property, borderPixel property, bottomPixel property, colorMap property, focusPixel property, foregroundPixel property, hiliteColor property, screenColors function, shadowPixel property, topPixel property, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies which entry in the color table is used for the background color when an object, or text in an object, is highlighted.

Syntax

set the hilitePixel of object to colorNumber

Example Code

```
set the hilitePixel of card button 8 to (the number of cards + 3)
```

Comments

Use the hilitePixel property to change the highlight color of an object when the bit depth of the screen is 8 bits (256 colors) or less.

Value:

The hilitePixel of an object is an integer between zero and (the screenColors - 1). It designates an entry in the current color table.

By default, the hilitePixel for all objects is empty.

Comments:

The hilitePixel property specifies which entry in the color table is used for an object's highlight color. It is similar to the hiliteColor property, but is specified as an entry in the current color table, rather than as a color reference.

The color table can be set by changing the colorMap property.

home

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

last keyword, middle keyword, number property, seconds function, two constant

Summary

Designates the application's main stack.

Syntax

Example Code

```
get the long name of stack home  
go home
```

Comments

Use the home keyword to find out where the main stack is.

Comments:

This keyword exists mainly to aid compatibility with imported HyperCard stacks.

In the development environment, the home stack is the License stack, and the statement go home does not do anything useful.

hostAddress

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

accept command, address function, hostAddressToName function, hostNameToAddress function, open socket command, openSockets function, peerAddress function

Summary

Returns the IP address of the local system a socket is connected to.

Syntax

the hostAddress of host:port[|connectionID]

hostAddress(host:port[|connectionID])

Example Code

```
the hostAddress of "www.example.com:80"  
hostAddress("example.net:25|10")  
the hostAddress of "127.0.0.1:8080|dataConnection"
```

Comments

Use the hostAddress function to find the IP address of the computer that Revolution is running on.

Parameters:

The host is an IP address or domain name.

The port is the number of the port the socket is connected to.

The connectionID is a string identifying the socket.

Value:

The hostAddress function returns the IP address of the computer. This address is in the form X.X.X.X, where each X is a number with between 1 and 3 digits.

Comments:

The socket must be open. If the specified socket has not been opened, the `hostAddress` function returns "not an open socket". If you have issued an open socket command to create the socket, you cannot use the `hostAddress` function until after the socket has been created and the command has completed.

The `connectionID` is needed only if more than one socket is connected to the same port of the same host. The `connectionID` is assigned by the `accept` or `open socket` command that created the socket.

hostAddressToName

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

accept command, hostAddress function, hostName function, hostNameToAddress function, open socket command, openSockets function

Summary

Returns the domain name corresponding to an IP address.

Syntax

the hostAddressToName of IPAddress

hostAddressToName(IPAddress)

Example Code

```
hostAddressToName("127.0.0.1")  
answer "Connecting to" && hostAddressToName(newAddress) & "..."
```

Comments

Use the hostAddressToName function to find the domain name of the computer at a IP address.

Parameters:

The IPAddress is a numeric IP address.

Value:

The hostAddressToName function returns the domain name corresponding to the IPAddress.

Comments:

The hostAddressToName function does a reverse DNS lookup on the IP address in order to find the domain name. This means that the computer must be connected to the Internet in order to use this function. If a connection to the Internet is not available, the hostAddressToName function returns an error message.

hostName

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

accept command, hostAddress function, hostNameToAddress function, open socket command, openSockets function, peerAddress function

Summary

Returns the domain name of the local system.

Syntax

the hostName

hostName()

Example Code

```
if the hostName is "localhost.example.org" then return empty
```

Comments

Use the hostName function to find the domain name of the computer that Revolution is running on.

Value:

The hostName function returns the domain name of the computer.

Comments:

The hostName function does a reverse DNS lookup on the hostAddress in order to find the domain name. This means that the computer must be connected to the Internet in order to use this function. If a connection to the Internet is not available, the hostName function returns an error message.

hostNameToAddress

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

accept command, hostAddress function, hostAddressToName function, hostName function, open socket command, openSockets function

Summary

Returns the IP addresses corresponding to a domain name.

Syntax

the hostNameToAddress of domainName

hostNameToAddress(domainName)

Example Code

```
hostNameToAddress("mail.example.org")  
put hostNameToAddress("www." & thisDomain) into line 3 of theHosts
```

Comments

Use the hostNameToAddress function to find the numeric IP address of the computer or computers at a given domain name.

Parameters:

The domainName is a standard fully-qualified domain name.

Value:

The hostNameToAddress function returns the IP addresses corresponding to the domain name, one per line.

Comments:

The hostNameToAddress function does a DNS lookup on the domain name in order to find the IP address. This means that the computer must be connected to the Internet in order to use this function. If a connection to the Internet is not available, the hostNameToAddress function returns an error message.

hotSpot

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

left property, mouseColor function, top property, xHot property, yHot property

Summary

Specifies the location of the hot spot when an image is being used as a cursor.

Syntax

set the hotSpot of image to point

Example Code

```
set the hotSpot of image "Arrow" to "20,0"
```

Comments

Use the hotSpot property to specify the position of the hot spot of an image you want to use as a cursor.

Value:

The hotSpot of an image is a point: an integer between 1 and the width of the image, a comma, and an integer between 1 and the height of the image.

By default, the hotSpot property of newly created images is set to "1,1".

Comments:

A mouse cursor, regardless of its shape, always has a single active point or hot spot. For example, the arrow cursor's hot spot is at its tip, and to select an object, you must click it with the arrow tip.

The first item of the hotSpot property is the horizontal distance in pixels from the left edge of the image to the hot spot point, and is equal to the image's xHot property. The second item of the hotSpot property is the vertical distance in pixels from the top of the image to the hot spot point, and is equal to the image's yHot property.

hotspotClicked

message

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

hotspots property, nodeChanged message, QTDebugStr message

Summary

Sent to a player containing a QuickTime VR movie when the user clicks one of the movie's hot spots.

Syntax

hotspotClicked hotspotID

Example Code

```
on hotSpotClicked theNode -- show a text description
  if theNode is among the lines of the clickableNodes of me then
    put line theNode of field "Tips" into field "Display"
  end if
end hotSpotClicked
```

Comments

Handle the hotspotClicked message if you want to perform some additional action when the user clicks a hot spot in a QuickTime VR movie.

Parameters:

The hotspotID is the ID of the QuickTime VR hot spot that was clicked.

Comments:

Each hot spot of a QuickTime VR movie is a clickable link, usually from one node to another. The movie author sets the hot spots during development of the movie. The user clicks a hot spot in the player to activate the link.

If the user navigates to another node by clicking a hot spot in the QuickTime VR movie, a nodeChanged message is sent after the hotspotClicked message.

hotspots

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

hotspotClicked message, nodes property, play command

Summary

Reports a list of the clickable hot spots in a QuickTime VR movie.

Syntax

get the hotspots of player

Example Code

```
get the number of lines in the hotspots of player "Tut's Tomb"
```

Comments

Use the hotspots property to list the clickable spots of a QuickTime VR movie.

Value:

The hotspots consists of a list of hot spots, one per line. Each line consists of two items, separated by a comma:

- the hot spot ID (an integer)
- the hot spot type ("link", "url", or "undefined")

The hotspots property is read-only and cannot be set.

Comments:

Each hot spot of a QuickTime VR movie is a clickable link, usually from one node to another. The movie author sets the hot spots during development of the movie. The user clicks a hot spot in the player to activate the link.

If the player does not contain a QuickTime VR movie, its hotspots property is empty.

hScroll

property

Synonyms

Objects

field, group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

formattedWidth property, hScrollbar property, vScroll property, width property

Summary

Specifies the horizontal scroll of a field or group.

Syntax

set the hScroll of {field | group} to pixels

Example Code

```
set the hScroll of field "Wasps" to the width of field "Wasp"
```

Comments

Use the hScroll property to scroll a field or group horizontally, or to check how far it's been scrolled.

Value:

The hScroll of a field, group, or stack is an integer between zero and the field's or group's formattedWidth.

By default, the hScroll property of newly created objects is set to zero.

Comments:

The hScroll is the amount in pixels the object has been scrolled to the right. If the hScroll is zero, the object has not been scrolled.

Setting the hScroll of a field or group causes a scrollbarDrag message to be sent.

hScrollbar

property

Synonyms

Objects

field, group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hScroll property, vScrollbar property, width property

Summary

Specifies whether a field or group has a horizontal scrollbar.

Syntax

set the hScrollbar of {field | group} to {true | false}

Example Code

```
set the hScrollbar of group "Main" to true
```

Comments

Use the hScrollbar property to show or hide the horizontal scrollbar of a field or group.

Value:

The hScrollbar of a field or group is true or false.

By default, the hScrollbar property of newly created fields or groups is set to false.

Comments:

When the hScrollbar of a group or field is set to true, the scrollbar appears across the bottom of the group or field.

Changing the hScrollbar does not change the object's rectangle property, so if there are objects near the bottom edge of the group, the scrollbar may cover them.

If a field's dontWrap property is set to true, the field does not scroll horizontally; the scrollbar appears but is always disabled.

HTMLText

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

charToNum function, colorNames function, foregroundColor property, format function, numToChar function, revPrintText command, RTFText property, text property, textFont property, textSize property, textStyle property, unicodeText property, URL keyword, How to copy styled text between fields, How to display a web page in a field, How to import and export styled text, How to store styled text in a variable or property, Recipe for changing underlines to italics in a field, Recipe for collecting text selections on the clipboard, Recipe for removing boldfacing from a field, Recipe for switching between styled text and HTML source, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies the contents of a field, with its text formatting represented as HTML tags and special characters represented as HTML entities.

Syntax

set the HTMLText of [chunk of] field to htmlString

Example Code

```
set the HTMLText of field "White Paper" to "<b><i>Click here!</i></b>"
set the HTMLText of field "Dest" to the HTMLText of field "Source"
write the HTMLText of field "Story" to file myWebFile
```

Comments

Use the HTMLText property to display text from a web page in a field, or copy or export a field's contents with the text formatting intact.

Value:

The HTMLText of a field or chunk is a string.

Comments:

The HTMLText property is a representation of the styled text of the field. Revolution uses a subset of HTML tags that includes font, size, style, and text color information.

Setting the HTMLText of a field (or a chunk of a field) sets both the text contents and the style attributes corresponding to the tags listed below. (Other tags are ignored.)

Getting the HTMLText property reports a string consisting of the text of the field (or chunk of a field), with any font, size, style, or color properties embedded in the text in the form of the tags listed below.

Note: The HTMLText of a field or chunk includes formatting information for the text, but does not include information about the text properties of the field itself. If you use the HTMLText property to transfer text between fields, you must make sure that the destination field's textFont and other text properties match the settings of the source field, if you want the text in both fields to look identical.

The tags translate as follows:

`<p> </p>`

Encloses a line of text. (Blank lines are also enclosed in `<p> </p>`.)

``

Encloses text whose textShift is a positive integer. (The `<sub>` tag is not nested for additional levels of subscription: it appears once for a run of subscripted text, regardless of the value of the textShift.)

``

Encloses text whose textShift is a negative integer. (The `<sup>` tag is not nested for additional levels of superscription: it appears once for a run of superscripted text, regardless of the value of the textShift.)

`<i> </i>`

Encloses text whose textStyle is "italic".

` `

Encloses text whose textStyle is "bold".

`<strike> </strike>`

Encloses text whose textStyle is "strikeout".

`<u> </u>`

Encloses text whose textStyle is "underline".

`<box> </box>`

Encloses text whose textStyle is "box".

`<threedbox> </threedbox>`

Encloses text whose textStyle is "threeDBox".

`<expanded> </expanded>`

Encloses text whose textStyle is "expanded".

`<condensed> </condensed>`

Encloses text whose textStyle is "condensed".

` `

Encloses text whose `textFont`, `textSize`, `foregroundColor`, or `backgroundColor` is different from the field's default. These five properties are represented as attributes of the `` tag.

• `face="fontName"` appears in the `` tag if the `textFont` is not the default.

• `size="pointSize"` appears if the `textSize` is not the default.

In standard HTML, the size attribute normally takes a value between 1 and 7, representing a relative text size, with 3 being the normal text size for the web page. To accommodate this convention, when setting the HTMLText of a field, if the `pointSize` is between 1 and 7, the `textSize` of the text is set to a standard value:

<code>pointSize</code>	<code>textSize</code>
1	8 point
2	10 point
3	12 point
4	14 point
5	17 point
6	20 point
7	25 point

• `lang="languageName"` appears if the `textFont` includes a language specification.

• `color="colorSpec"` appears if the `foregroundColor` is not the default.

• `bgcolor="colorSpec"` appears if the `backgroundColor` is not the default.

When getting the `htmlText` of a field, a color is represented as an HTML-style color consisting of a hash mark (#) followed by three 2-digit hexadecimal numbers, one for each of red, green, and blue.

`<a> `

Encloses text whose `textStyle` is "link" or whose `linkText` property is not empty. If the `textStyle` of the text contains "link", the `linkText` is included as the value of the "href" attribute. Otherwise, it is included as the value of the "name" attribute.

``

Replaces a character whose `imageSource` property is not empty. The value of the `imageSource` property is included as the value of the "src" attribute.

When you set the HTMLText of a field, all tags other than those above are ignored, except heading tags (`<h1>ó<h6>`), which change the size of the text in the heading element:

<code>tag</code>	<code>textSize</code>
<code><h1></code>	34 point
<code><h2></code>	24 point
<code><h3></code>	18 point
<code><h4></code>	14 point
<code><h5></code>	12 point
<code><h6></code>	10 point

You can use Revolution color references for the "color" and "bgcolor" attributes. Revolution translates these into standard HTML-style color specifications.

If a chunk of text includes more than one of the above styles, Revolution encloses the text in the tags corresponding to each style, from the inside out. For example, if the word "Flail" in a field is underlined and bold, its corresponding HTMLText is "`<u>Flail</u>`".

Special characters (whose ASCII value is greater than 127) are encoded as HTML entities. Revolution recognizes the following named entities:

í	Á
·	á
¬	Â
,	â
¥	´
Δ	Æ
Ê	æ
ì	À
‡	à
≈	Å
Â	å
√	Ã
„	ã
f	Ä
‰	ä
¶	¦
«	Ç
Á	ç
Π	¸
¢	¢
©	©
§	¤
∞	°
~	÷
...	É
È	é
	Ê
‘	ê
»	È
Ë	è
—	Ð
□	ð
À	Ë
"	ë
Ω	½
°	¼
æ	¾
>	>
Õ	Í
"	í
Œ	Î
Ó	î
°	¡
Ã	Ì
Ï	ì
ø	¿

œ	&luml;
Ô	ï
'	«
<	<
Ø	¯
μ	µ
Σ	·
†	
..	¬
—	Ñ
Ò	ñ
”	Ó
Û	ó
‘	Ô
Ù	ô
“	Ò
Ú	ò
™	ª
∫	º
ÿ	Ø
̲	ø
'	Õ
ı	õ
÷	Ö
^	ö
∂	¶
±	±
£	£
ª	»
Æ	®
ß	§
≠	­
π	¹
≤	²
≥	³
fl	ß
fi	Þ
ˆ	þ
◊	×
/	Ú
·	ú
€	Û
°	û
ÿ	Ù
˘	ù
®	¨
◁	Ü
,	ü
›	Ý

” ý
• ¥
˘ ÿ

Unicode characters whose numeric value is greater than 255 are encoded as "bignum" entities, with a leading ampersand and trailing semicolon. For example, the Japanese character whose numeric value is 12387 is encoded as "っ".

Important! The HTMLText property uses a tag structure that is HTML-like, but is not completely standard HTML, in order to accommodate the full range of text styling available in Revolution. Specifically:

• The <link>, <box>, <threedbox>, <expanded>, and <condensed> tags, as well as the bgColor attribute of the tag, have been added to accommodate styles that don't exist in standard HTML.

• The size attribute of the tag can encode the font's point size, in addition to the standard 7 HTML sizes.

• The HTMLText reports entities whose ASCII value is between 129 and 159. These correspond to characters in the Windows character set (code page 1252) that are not legal HTML entities.

Changes to Transcript:

The lang attribute, and the ability to encode Unicode characters, was added in version 2.0.

The use of the <a> tag to enclose text whose textStyle property is "link" was introduced in version 1.1.1. In previous versions, <link> and </link> enclosed text whose textStyle was "link" and whose linkText property was empty.

In versions before 1.1, tags were capitalized. (Tags are not case-sensitive, so when you set the HTMLText property, you can use either uppercase or lowercase.)

The <a> tag was introduced in version 1.1. In previous versions, the <GROUP> tag was used instead.

In version 1.1, the behavior of the <p> tag changed. In previous versions, the <P> tag was interpreted as a line-break character. In version 1.1, <p> and </p> enclose each line.

The translation between the linkText property and the <a> tag, and between the imageSource property and the tag, was introduced in version 1.1. In previous versions, these two properties were not available in the HTMLText.

http
keyword

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

binfile keyword, file keyword, ftp keyword, httpHeaders property, libURLErrorData function, libURLSetCustomHTTPHeaders command, load command, resfile keyword, URL keyword, URLStatus function, About using URLs, uploading, and downloading, How to access the Internet from behind a firewall, How to display a web page in a field, How to open a URL in a browser, Why can't I upload a file?, Why don't URLs work in a standalone?, Why was a downloaded file corrupted?

Summary

Used as a URL type with such commands as put and get to designate a file on the World Wide Web.

Syntax

Example Code

```
set the htmlText of field 1 to URL "http://example.org/data.html"  
put URL "http://www.example.com/output?this=that" into testData  
put line 2 of URL "http://www.example.com/stuff/" into testDate
```

Comments

Use the http keyword to work with files on the Web.

Comments:

The URL scheme "http" indicates information located on a web server. An http URL consists of:

1. The string "http://"
2. An optional user name and password, separated by a colon (:) and followed by "@"
3. The name of the server (for instance, "example.net") followed by a slash
4. The location of the resource (often, a file path).

Here are some examples of valid http URLs:

http://www.example.com

The main page for the server "www.example.com"

`http://www.example.com/directory/`
A directory on the server

`http://www.example.com/directory/file.html`
A file on the server

`http://user:password@www.example.com/file.txt`
A file accessed by a user name and password

`http://www.example.com/directory/stuff.html?list=yes`
A page generated by a queryópossibly generated by a CGI

Important! If your user name or password contains any of the characters ":", "@", "/", ".", or "|", use the `URLEncode` function to safely encode the user name or password before putting them into the URL. The following example constructs a URL for a user whose password contains the "@" character:

```
put "name" into userName
put "jdoe@example.com" into userPassword
put "http://" & userName & ":" & URLEncode(userPassword)
```

```
& "@www.example.net/index.html" into fileURLToGet
get URL fileURLToGet
```

An http URL is a container, and you can use the expression `URL httpURL` in any statement where any other container type is used. When you get the value of an http URL, Revolution downloads the URL from the server. (If you have previously cached the URL with the `load` command, it fetches the URL from the cache.)

If an error occurs during transfer of the data, the error is placed in the result function. The first word returned by the result function is "error", followed (where appropriate) by the text of the error message returned by the HTTP server, including the server response code.

Important! If there is an error downloading an http URL, the URL container does not necessarily evaluate to empty. Most HTTP servers send an error page when the file is not found or another error occurs, and the URL container will evaluate to the contents of this page. Before using the data in a URL container, check the result to make sure it is empty and there was no error.

You can upload data to a web server by putting a value into an http URL, as in the following statement:

```
put field "Info" into URL "http://www.example.net/info.html"
```

However, because most web servers do not allow HTTP uploads, the attempt usually will not be successful. (Check with the server's administrator to find out how and where to upload files.)

Note: Transferring a URL by using it in an expression is a blocking operation: that is, the handler pauses until Revolution is finished getting the URL. Since contacting a server may take some time due to network lag, URL operations may take long enough to be noticeable to the user.

Important! If a blocking operation involving a URL (using the put command to upload a URL, the post command, the delete URL command, or a statement that gets an ftp or http URL) is going on, no other blocking URL operation can start until the previous one is finished. If you attempt to use a URL in an expression, or put data into a URL, while another blocking URL operation is in progress, the result is set to "Error Previous request not completed".

Downloading a URL does not prevent other messages from being sent during the download: the current handler is blocked during the download, but other handlers are not. This means that if, for example, your application has a button that downloads a URL, the user might click the button again (or click another button that downloads another URL) while the first URL is still being downloaded. In this case, the second download is not performed and the result is set to "error Previous request has not completed." To avoid this problem, you can set a flag while a URL is being downloaded, and check that flag when trying to download URLs to make sure that there is not already a current download in progress.

The following example shows how to set a flag in a global variable to prevent multiple downloads. The variable "downloadInProgress" is set to true while a download is going on, and back to false when the download concludes. If the user clicks the button again while the download is still going on, the handler simply beeps:

```
on mouseUp
  global downloadInProgress
  if downloadInProgress then
    beep
  exit mouseUp
end if
put true into downloadInProgress -- about to start
put URL (field "Page to get") into field "Command Result"
put false into downloadInProgress -- finished
end mouseUp
```

For technical information about URLs and the http URL scheme, see RFC 1630 at <http://www.ietf.org/rfc/rfc1630.txt>.

Important! The http keyword is part of the Internet library. To ensure that the keyword works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the http keyword is not yet available. This may affect attempts to use this keyword in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the http keyword can be used in any handler.

Changes to Transcript:

The http keyword was moved to the Internet library in version 1.1. In previous versions, this functionality was part of the engine.

The ability to use authenticated http URLs (with a user name and password) was introduced in version 1.1.1.

httpHeaders

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

libURLSetCustomHTTPHeaders command, load command, post command, revGoURL command, URL keyword, URLEncode function, How to access the Internet from behind a firewall

Summary

Specifies custom headers to be sent with each GET or POST request to an HTTP server.

Syntax

set the httpHeaders to headersList

Example Code

```
set the httpHeaders to field "Headers" & return & field "Special"
```

Comments

Use the httpHeaders property to supply custom headers when interacting with HTTP servers that require them.

Value:

The httpHeaders is a string.

By default, the httpHeaders property is set to empty.

Comments:

Whenever Revolution contacts a web server to download a page (with the load command or by using a URL in an expression) or to post data (with the post command), the contents of the httpHeaders property is sent to the web server along with the default headers.

The custom header lines specified by the httpHeaders are sent along with a set of default headers. If the headersList includes any header lines that are part of the default headers, the one in the headersList replaces the default header. Any new lines are appended to the end of the headers to be sent to the server.

To replace the default headers instead of adding to them, use the `libURLSetCustomHTTPHeaders` command instead.

Important! If you have used the `libURLSetCustomHTTPHeaders` command to set all the headers, the `httpHeaders` setting is ignored and the headers set by `libURLSetCustomHTTPHeaders` are used instead.

For technical information about the standard headers recognized in the HTTP 1.1 protocol, see RFC 2616 at <http://www.ietf.org/rfc/rfc2616.txt>.

HTTPProxy

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

accept command, http keyword, httpHeaders property, load command, open socket command, post command, URL keyword, How to access the Internet from behind a firewall

Summary

Specifies a proxy server to be used for HTTP requests.

Syntax

set the HTTPProxy to host:portNumber

Example Code

```
set the HTTPProxy to "127.0.0.0:80"
```

Comments

Use the HTTPProxy property to use HTTP servers from behind a firewall.

Value:

The HTTPProxy consists of a host and port number, separated by a colon. The host is the IP address of the proxy server. The portNumber is the network port to send requests to. (Generally, port 80 is used for HTTP requests.)

Comments:

Normally, when an application requests a web document, the request goes directly to the HTTP server that hosts the document. If your computer is behind a network firewall or if the HTTP server is blocked in some other way, the request must be sent to a proxy server, which fetches the document from the host and then pass it back to your system.

If the HTTPProxy property is empty, requests are sent directly to the host server and no proxy is used. This is the default setting. (On Windows systems, if a proxy server is set in the registry, that setting is used as the default.)

iBeam

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command, cursor property, nine constant

Summary

Equivalent to the number 9.

Syntax

Example Code

```
set the cursor to iBeam
```

Comments

Use the iBeam constant to set the cursor to an I-beam, suitable for placing a text insertion point in a field.

Comments:

The following two statements are equivalent:

```
set the cursor to iBeam  
set the cursor to 9
```

However, the first is easier to read and understand in Transcript code.

Important! If you use the iBeam cursor or other standard cursors in your application, you must include it when you create your standalone. Make sure the "Cursors" option on the Inclusions tab in Step 3 of the Distribution Builder is checked.

icon

property

Synonyms

Objects

global, button, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

armedIcon property, disabledIcon property, hilitedIcon property, iconic property, iconify message, showIcon property, visitedIcon property, How to display an icon or picture in a button, Why do icons disappear from a standalone application?, Development menu > Image Library

Summary

Specifies an image that is displayed in a button, or used as the desktop icon of a stack file or application.

Syntax

set the icon to {imageID | imageName}

set the icon of {button | stack} to {imageID | imageName}

Example Code

```
set the icon of this stack to 974
set the icon of button "Help" to "Question Mark"
set the icon to the myAppIcon of stack "Main Settings"
```

Comments

Use the icon property to change a button's appearance, or to set the icon used for a stack or application in the OS X dock.

Value:

The icon is the short ID or short name of the image to use for the current application's dock icon. The icon of a button or stack is also a short name or ID of an image.

By default, the icon property is set to zero (no icon). The icon of newly created buttons and stacks is set to zero (no icon) by default.

Comments:

Revolution looks for the specified image first in the current stack, then in other open stacks.

If a button has been clicked during the current session and its `visitedIcon` property is set, its `visitedIcon` is displayed instead of its icon. If the button is disabled and its `disabledIcon` property is set, its `disabledIcon` is displayed instead of its icon.

Cross-platform note: On OS X systems, a stack's icon is displayed in the dock when the stack is minimized. On Unix systems, the stack's icon is displayed on the desktop when the stack is iconified. Setting a stack's icon property has no effect on Mac OS and Windows systems.

Cross-platform note: On OS X systems, the global icon property specifies the current application's dock icon. The setting of the global icon property has no effect on Mac OS, Unix, or Windows systems.

Changes to Transcript:

The ability to specify an icon for an application or stack file on OS X systems was added in version 2.1. In previous versions, the icon was a button and stack property but not a global property, and the icon of a stack had no effect on OS X systems.

iconic property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

close command, collapseBox property, iconifyStack message, minimize keyword, startUpIconic property

Summary

Specifies whether a stack window is minimized.

Syntax

set the iconic of stack to {true | false}

Example Code

```
set the iconic of this stack to true
```

Comments

Use the iconic property to change a stack's appearance in order to make it take up less screen space.

Value:

The iconic of a stack is true or false.

By default, the iconic property of newly created stacks is set to false.

Comments:

Set a stack's iconic property to true to make the stack window take up less space. The way in which this is done depends on the platform:

- On Mac OS systems, the window is collapsed to its title bar
- On OS X systems, the window is collapsed to an icon in the Dock
- On Unix systems, the window is iconified into a desktop icon
- On Windows systems, the window is minimized to an icon in the task bar

If the user has collapsed, iconified, or minimized the stack, its iconic property reports true.

Changes to Transcript:

Support for setting the icon of a stack on OS X systems was added in version 2.0.

iconifyStack

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeStack message, collapseBox property, decorations property, icon property, iconic property, resizeStack message, startupIconic property, suspendStack message, unIconifyStack message

Summary

Sent to the current card when a stack is minimized.

Syntax

iconifyStack

Example Code

```
on iconifyStack -- hide auxiliary window when the stack is iconified
    hide stack "Go Back Palette"
end iconifyStack
```

Comments

Handle the iconifyStack message if you want to do something special when the user minimizes a stack window.

Comments:

The terminology varies depending on platform. The iconifyStack message is sent when the user collapses (Mac OS systems), iconifies (Unix systems), or minimizes (OS X systems and Windows systems) the stack window.

ID

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

abbreviated keyword, altID property, cardIDs property, groupIDs property, IDChanged message, long keyword, name property, number property, short keyword, About object types and object references, How to refer to a control on another card

Summary

Reports the unique ID number assigned to an object.

Syntax

set the ID of {image | stack} to number

get the [long | abbr[ev[iated]] | short] ID of object

Example Code

```
send mouseUp to button ID 2214
set the ID of image "Custom Cursor" to 2314
put the long ID of this card into savedID
```

Comments

Use an object's ID property to refer to the object in an unambiguous way.

Value:

The ID of an object is a non-negative integer.

Comments:

A stack's ID is equal to the ID that will be assigned to the next object created within that stack, so the stack ID is subject to change. You can set the ID of a stack, but only to a greater number than its current ID.

You can set the ID of an image. Be careful not to set an image ID to a number that's the ID of another object in the same stack: since Revolution uses IDs to keep track of objects, a conflict may prevent Revolution from being able to access one or both objects. The following ID numbers are reserved and should not be used for image IDs:

- ï 1-100: reserved for built-in cursors
- ï 101-135: reserved for built-in brush shapes
- ï 236-300: reserved for built-in patterns
- ï 301-1000: reserved for built-in icons
- ï 101,000-103,000: reserved
- ï 200,000-299,999: reserved for application use

For all other objects, the ID property is assigned when the object is created and never changes. This means that for all objects except stacks and images, the ID property is guaranteed to be persistent. An object's name or number may change, but its ID does not.

For all objects, the ID is guaranteed to be unique within a stack. IDs are not reused if the object is deleted.

Note: An object that is created by copying and pasting, or with the copy command, is assigned a new ID. If you cut an object and paste it into the same stack, it retains its original ID.

The short ID of an object is its ID number. If you don't specify a modifier for the ID property, you get the short ID form.

The abbreviated ID of an object is the object's type, followed by "id", followed by the object's short ID. For example, if a button's short ID is "27", its abbreviated ID is "button id 27".

The long id of an object includes information about its owner (and about the owner of that object, and so forth). For example, suppose a stack named "My Stack" contains a card whose ID is 11. This card has a group whose ID is 28, which in turn contains a button whose ID is 34. The card also has a card field whose ID is 46. If "My Stack" is a main stack and it's in a file whose path is "/Drive/Folder/Stack.rev", the long IDs of these objects look like this:

- ï The stack: stack "/Drive/Folder/Stack.rev"
- ï The group: group id 2238 of card ID 1001 of stack "/Drive/Folder/Stack.rev"
- ï The card: card id 1001 of stack "/Drive/Folder/Stack.rev"
- ï The grouped button: button id 34 of group id 2238 of card id 1001 of stack
"/Drive/Folder/Stack.rev"
- ï The card field: field id 46 of card id 1001 of stack "/Drive/Folder/Stack.rev"

If the stack is a substack, its ID is included in the long name of each of its objects, before the path of the main stack.

The long ID of a group includes the ID of the current card. If the group does not appear on the current card, requesting its ID causes an execution error. If you need to get the ID of a group, use the "background" terminology instead.

The long ID of a background includes the ID of the current card, if the background appears on the current card. If not, the long ID of the background includes the ID of the first card the background appears on.

If an object's name is empty, getting its name yields its ID property instead.

Important! If a stack was originally created with HyperCard and then imported into Revolution, the ID of each control in the stack is guaranteed unique only within its domain. (You can check a stack's HCStack property to determine whether it began life in HyperCard.)

Note: If a stack's HCAAddressing property is set to true, the long or abbreviated ID of a control in that stack begins with the word "background" if the control is part of a group, and with the word "card" if not.

IDChanged

message

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

altID property, ID property, nameChanged message, number property

Summary

Sent to an image when its ID property is changed.

Syntax

IDChanged oldID,newID

Example Code

```
on IDChanged prevID,currentID -- update ID stored in another object
    set the storedID[the short name of image ID prevID] of this stack
        to currentID
end IDChanged
```

Comments

Handle the IDChanged message if you want to make updates when an image's ID is changed.

Parameters:

The oldID is the image's original ID number.

The newID is the image's new ID number.

Comments:

The ID property of most objects is set when the object is created and cannot be changed. There are two exceptions: stacks and images.

You can set the ID of an image to any positive integer. Be careful not to set an image ID to a number that's the ID of another image in the same stack: since Revolution uses IDs to keep track of objects, a conflict may result in the inability to access one or both objects. The following ID numbers are reserved and should not be used for image IDs:

ï 1-100: reserved for built-in cursors

- ï 101-135: reserved for built-in brush shapes
- ï 236-300: reserved for built-in patterns
- ï 301-1000: reserved for built-in icons
- ï 101,000-103,000: reserved
- ï 200,000-299,999: reserved for application use

The actual change is not triggered by the IDChanged message, so trapping the message and not allowing it to pass does not prevent the ID from being changed.

idle

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

idleRate property, revVideoGrabIdle command, send command

Summary

Sent periodically to the current card if no other message is being sent.

Syntax

idle

Example Code

Comments

Handle the idle message to check constantly on the status of an object, the content of a variable, and so forth, or to do periodic tasks.

Comments:

The period between idle messages is specified by the idleRate and idleTicks properties.

Note: Usually, it is easier and more efficient to use the send in time form of the send command than to use an idle handler, especially if a task needs to be executed at regular intervals. This example shows an idle handler that updates a clock timer:

```
on idle -- avoid if possible
  global startTime
  if the seconds > startTime + 60 -- 60 seconds have gone by
    put the time into field "Clock Face"
    put the seconds into startTime
  end if
  pass idle
end idle
```

The following example does the same thing more efficiently, since it only needs to handle a single message every sixty seconds:

```
on updateClock -- a better way
  put the time into field "Clock Face"
  send "updateClock" to me in 60 seconds
end updateClock
```

Executing an idle handler slows down other Revolution actions, so handle the idle message only in the rare cases where the send command cannot be used instead.

Note: If there is no idle handler anywhere in the message path, no idle message is sent.

idleRate

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

effectRate property, idle message, idleTicks property, syncRate property

Summary

Specifies the number of milliseconds between idle, mouseStillDown, and mouseWithin messages.

Syntax

set the idleRate to number

Example Code

```
set the idleRate to 500 -- one idle message per half second
```

Comments

Adjust the idleRate property to change the interval between periodically-sent messages. Increasing the idleRate causes these messages to be sent less frequently, and decreases the amount of CPU time the application uses.

Value:

The idleRate is an integer between zero and 65535.

By default, the idleRate property is set to 200 (one-fifth of a second).

Comments:

The idleRate is the time in milliseconds between one idle message and the next, one mouseStillDown message and the next, and one mouseWithin message and the next.

Important! Some Unix systems cannot reliably time an interval less than 200 milliseconds. Setting a shorter idleRate on those systems may cause idle messages to be sent at erratic times.

This property is very similar to the idleTicks property, and changes when the idleTicks changes. The only difference is that the two properties are given in different time units: the idleRate is in milliseconds and the idleTicks is in ticks.

idleTicks

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

idle message, idleRate property, ticks function

Summary

Specifies the number of ticks between idle, mouseStillDown, and mouseWithin messages.

Syntax

set the idleTicks to number

Example Code

```
set the idleTicks to 60 -- 1 second interval
```

Comments

Adjust the idleTicks property to change the interval between periodically-sent messages. Increasing the idleTicks causes these messages to be sent less frequently, and decreases the amount of CPU time the application uses.

Value:

The idleTicks is a positive integer.

By default, the idleTicks property is set to 12 (one-fifth of a second).

Comments:

The idleTicks is the time in ticks between one idle message and the next, one mouseStillDown message and the next, and one mouseWithin message and the next.

Important! Some Unix systems cannot reliably time an interval less than 12 ticks. Setting a shorter idleTicks on those systems may cause idle messages to be sent at erratic times.

This property is very similar to the idleRate property, and changes when the idleRate changes. The only difference is that the two properties are given in different time units: the idleRate is in milliseconds and the idleTicks is in ticks.

if
control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

else keyword, end if keyword, switch control structure, then keyword

Summary

Executes a list of statements if a condition is true.

Syntax

if condition then statement [else elseStatement]

Example Code

Comments

Use the if control structure to execute a statement (or list of statements) only under certain circumstances.

Form:

The if control structure always begins with the word if. There are four forms of the if control structure:

if condition then statement [else elseStatement]

(This form may have a line break before the words then or else or both.)

```
if condition then
    statementList
[else
    elseStatementList]
end if
```

```
if condition
then statement
[else
    elseStatementList]
end if]
```

```
if condition then
  statementList
else elseStatement
```

Parameters:

The condition is any expression that evaluates to true or false.

The statementList or elseStatementList consists of one or more Transcript statements, and can also include if, switch, try, or repeat control structures. The statement or elseStatement consists of a single Transcript statement.

Comments:

If the condition evaluates to true, the statement or statementList is executed; if the condition evaluates to false, the statement or statementList is skipped.

If the if control structure contains an else clause, the elseStatement or elseStatementList is executed if the condition is false.

If one if control structure is nested inside another, use of the second form described above is recommended, since the other forms may cause ambiguities in interpreting which else clause belongs with which if statement.

The if control structure is most suitable when you want to check a single condition. If you need to check for multiple possibilities, doing something different for each one, use a switch control structure instead.

Note: The if control structure is implemented internally as a command and appears in the commandNames.

image
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

button keyword, choose command, field keyword, graphic keyword, player keyword, scrollbar keyword, style property, templateImage keyword, tool function

Summary

Designates the Image tool, which is used to create new paint images.

Syntax

Example Code

```
choose image tool
```

Comments

Use the image keyword to create images.

Comments:

When using the Image tool, the cursor is a crosshairs. This tool is used for creating images by clicking at one corner of the image and dragging to the opposite corner.

image object

Synonyms
img

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

graphic object, imageData property, templateImage keyword, About object types and object references, How to take a picture with a video camera, File menu > Import As Control > Image File..., File menu > New Referenced Control > Image File..., Object menu > New Control > Image

Summary

A control that contains a bitmapped picture.

Syntax

Example Code

```
hide image "Blinking Star"  
export image ID 9234 to file "Test Image.jpg"
```

Comments

Use the image object type to hold photographs, icons, and decorative elements, and to allow the user to paint.

Comments:

Each image contains a bitmapped picture, which can either be imported with the import command or created using the paint tools. An image can display either its own data, or a file specified by the image's filename property.

An image is contained in a card, group, or background. Images cannot contain other objects.

The image object has a number of properties and messages associated with it. To see a list of messages that can be sent to an image as a result of user actions or internal Revolution events, open the "Transcript Language Dictionary" page of the main Documentation window, and choose "Image Messages" from the Show menu at the top. To see a list of all the properties an image can have, choose "Image Properties" from the Show menu.

imageData

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

alphaData property, crop command, filename property, flip command, maskData property, revVideoFrameImage function, rotate command, templateImage keyword

Summary

Specifies the binary data that makes up the picture in an image object.

Syntax

set the imageData of image to binaryData

Example Code

```
put the imageData of image ID 3577 into dateToAnalyze
set the imageData of the mouseControl to the personalImage of this card
```

Comments

Use the imageData property to process an image and display the processed version.

Value:

The imageData of an image consists of a sequence of binary values.

Comments:

The imageData consists of the picture data presented in a standard form. The form of the imageData property, unlike the content, does not depend on what format the image is recorded in; it's always in the same form, which specifies the color of each pixel in the image, four bytes per pixel.

Each pixel is represented by 32 bits (4 bytes) of image data, with pixels numbered from the top left corner of the image, left to right, then top to bottom. The first byte consists of zeroes, and the last three bytes encode the amount of red, green, and blue respectively.

Since each pixel is represented by 4 bytes (4 characters), you can obtain the numeric value of any of the color channels for a given pixel using the charToNum function. For example, the numeric value of the red channel for the tenth pixel is given by the expression `charToNum(char ((4 * 9) + 2))` of the imageData of image). The numeric value of the green channel is `charToNum(char (4 * 9) + 3)` of the

imageData of image); and the numeric value of the blue channel is `charToNum(char (4 * 9) + 4 of the imageData of image)`.

Important! When changing the `imageData` property, make sure the new data is the correct size: 4 bytes per pixel in the image. If you set an image's `imageData` property to data whose total length is incorrect, the image appearance will be distorted.

The `imageData` property is related to the content of the image, but changing either one changes what's displayed in the image, but they're not identical: the `imageData` property and the image content are in different forms, have different sizes, and include overlapping but not identical information about the picture.

The `imageData`, unlike the contents of the image container, is based on the picture as it's presented on the screen, not stored in the image object. This means that if you resize an image, the content of the image does not change, but its `imageData` does. If you create an image and then reduce its size, its `imageData` reflects the scaled-down, displayed image, not the original full-scale image. If you create a second image and set its `imageData` property to the `imageData` of the original image, resizing the first image back to the original dimensions displays the original image at full resolution, but resizing the second image does not, because setting its `imageData` transferred only the scaled-down version of the original.

Tip: To copy the information in an image at full resolution, regardless of whether its height and width have been changed, use a statement like the following:

```
put image "Full Resolution" into image "Copied Image"
```

Important! Since the `imageData` of an image is binary data rather than text, trying to display the data in a field may cause unexpected behavior.

imagePixmapID

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

externals property, maskPixmapID property, pixmapID property, windowID property

Summary

Specifies the internal window ID of the pixmap used to redraw an image.

Syntax

get the imagePixmapID of image

Example Code

```
put the imagePixmapID of image "Flower" into myStoredID
if the imagePixmapID of image 6 is not the storedID of me then beep
```

Comments

Use the imagePixmapID property to pass the window ID to an external that needs to manipulate the window.

Value:

The imagePixmapID of a stack is an integer.

This property is read-only and cannot be set.

Comments:

The pixmap ID is provided by the operating system.

Some externals need to manipulate the contents of the image directly, and do so by referencing the ID. Such externals require you to pass the ID when you use the external.

imageSource

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

character keyword, hide command, show command

Summary

Specifies an image to be displayed in place of the specified character in a field.

Syntax

set the imageSource of character to {imageID |imageName |imageURL |empty}

Example Code

```
set the imageSource of char 1 of line 2 of field "This" to 2533
set the imageSource of char thisChar of field 1 to "Arrow"
set the imageSource of last char of me to "binfile:My Image"
```

Comments

Use the imageSource property to display a picture or icon inside a text field.

Value:

The imageSource of a character is either empty or an image specifier.

An imageID is the ID of an image to display instead of the character. Revolution looks for the specified image first in the current stack, then in other open stacks.

An imageName is the short name of an image to display instead of the character. Revolution looks for the specified image first in the current stack, then in other open stacks.

An imageURL is any valid URL that specifies an image in a format Revolution can display.

By default, the imageSource for all characters is empty.

Comments:

Setting the imageSource of a character hides the character.

Setting the `imageSource` of a character to empty removes the image and allows the character to appear instead.

If the image is noticeably taller than the text in the field, it may appear cut off if the field's `fixedLineHeight` property is set to true. To correct this problem, set the field's `fixedLineHeight` to false.

Important! Don't use the URL keyword when specifying an `imageURL`. The `imageSource` of a character is a file location (or image ID or name), not the image data itself. If you use the URL keyword, the `imageSource` property is set to the contents of the URL, not the URL itself, and this is usually not what's wanted.

import

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

defaultStack property, export command, filename property, import snapshot command, play command, Image Types Reference, How to display a TIFF file, How to display an icon or picture in a button, How to import a picture file into an existing image object, How to import a text file into a field, How to import a Unicode text file, How to import an RTF file, How to import and export styled text, Recipe for getting the dimensions of a picture file, File menu > Import As Control

Summary

Creates an image, EPS, audio clip, or video clip and copies the contents of a file into the object.

Syntax

import type from file filePath [with mask maskFilePath]

Example Code

```
import paint from file "flowers.jpeg"  
import audioClip from file "/sounds/temp/bugsounds.wav"  
import paint from file it with mask "/etc/res/oval.pbm"
```

Comments

Use the import command to place a sound, movie, or image in a stack, instead of creating a referenced control that uses an external file.

Parameters:

The type is one of paint, EPS, audioClip, or videoClip.

The filePath specifies the name and location of the file you want to import from. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

The maskFilePath specifies the name and location of a one-bit mask file for an image. You can use a mask file only if the file being imported is in the PBM format.

Comments:

The imported object is placed in the current stack. To change the current stack before importing, set the defaultStack property to the stack you want to import into, then import the file.

Importing a paint file creates an image object on the current card. The import command can import GIF, JPEG, PNG, BMP, XWD, XBM, XPM, or PBM, PGM, or PPM files. On Mac OS systems, PICT files can also be imported (but they cannot be displayed on Unix or Windows systems).

Importing an EPS file creates an EPS object on the current card.

Importing an audioClip or videoClip file creates an audio clip or video clip in the current stack. Audio files can be in WAV, AIFF, or AU format. Video files can be in QuickTime, AVI, or MPEG format.

Note: The import command places the data in the stack file itself. This assures that the information stays with the stack if it's moved, but it also increases the stack's size and the amount of memory used by the stack. If you prefer to keep the data outside the stack file, you can display a picture file without importing it by creating an image and setting its filename property, or by choosing File menu New Referenced Control Image File. To allow playing a sound or movie without importing it, create a player and set its filename property, or choose File menu New Referenced Control QuickTime-Supported File.

import snapshot

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

export snapshot command, import command, rectangle property, screenRect function, windowID property, How to create a screenshot of a stack, File menu > Import As Control > Snapshot

Summary

Creates an image of a portion of the screen.

Syntax

import snapshot [from rect[angle] rectangle [of window windowID]]

Example Code

```
import snapshot -- displays crosshairs to select an area
import snapshot from rectangle 100,100,500,400
import snapshot from rectangle (the rect of group "Picture")
import snapshot from rect myRect of window 91373124
```

Comments

Use the import snapshot command to place a screenshot in the current stack.

Parameters:

The rectangle specifies the edges of the rectangle to be imported, separated by commas, in the same order as the rectangle property (left, top, right, bottom).

Important! If a window is specified, the rectangle is given in relative (window) coordinates; otherwise, it is given in absolute coordinates.

The windowID is the windowID property of an open stack window. (This is not the same as the stack's ID property.)

Comments:

The import snapshot command creates a new image in the center of the current card and places the snapshot in the image.

If you don't specify a rectangle, Revolution displays a crosshairs cursor. Click at one corner of the rectangle to be imported and drag to the opposite corner to select the area.

If you specify a `windowID`, the rectangle's coordinates are relative to the top left corner of the window you specify. However, if the window is partly overlapped by another window, whatever is visible on the screen within that rectangle is placed in the snapshot. In other words, you cannot take a snapshot of a part of a window that is hidden by another overlapping window.

While you take the snapshot, Revolution hides its own windows (such as the Tools palette).

The format of the resulting image depends on the current setting of the `paintCompression` property.

in
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

of keyword, the keyword

Summary

Designates the string that contains a chunk expression.

Syntax

Example Code

```
repeat with x = 1 to the number of items in the hilitedLines of me
```

Comments

Use the in keyword as a synonym for the of keyword in chunk expressions.

Comments:

When designating a part of a string, use the in keyword after the chunk type. For example, to specify the first word of a string, use an expression like this:

word 1 in string

In complex chunk expressions with multiple chunk types, use the in keyword after each chunk type. For example, to specify the last character of the second line of a string, use an expression like this:

last char in line 2 in string

When designating the number of chunks in a string, use the in keyword after the chunk type. For example, to get the number of lines in a string, use an expression like this:

the number of lines in string

ink

property

Synonyms

Objects

button, field, graphic, image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backgroundColor property, blendLevel property, foregroundColor property, layer property, opaque property, screenDepth function, Shortcut to make an image transparent

Summary

Specifies the kind of transparency an object is drawn with.

Syntax

set the ink of object to inkMode

Example Code

```
set the ink of graphic "Floating" to reverse
set the ink of field "Ghost" to noOp
```

Comments

Use the ink property to specify how the color of each pixel of an object combines with the color of the pixel underneath it to create the final color shown on the screen.

Value:

The ink of an object is a string designating one of 26 transfer modes.

By default, the ink of newly created objects is "srcCopy".

Comments:

The srcCopy mode simply puts the object on top of the background; the background does not show through or affect the final color at all.

You can use other transfer modes with the ink property to create special transparent or partially transparent effects, or to invert the colors of an object or the colors underneath the object, or to combine the object's colors with the colors under the object.

There are 26 transfer modes:

- `addMax` (Mac OS and OS X only)
- `adMin` (Mac OS and OS X only)
- `addOver` (Mac OS and OS X only)
- `addpin` (Mac OS and OS X only)
- `blend`
- `clear`
- `noop`
- `notSrcAnd`
- `notSrcAndReverse`
- `notSrcBic` (Mac OS and OS X only)
- `notSrcCopy`
- `notSrcOr`
- `notSrcOrReverse`
- `notSrcXor`
- `reverse`
- `set`
- `srcAnd`
- `srcAndReverse`
- `srcBic` (Mac OS only)
- `srcCopy`
- `srcOr`
- `srcOrReverse`
- `srcXor`
- `subOver` (Mac OS and OS X only)
- `subPin` (Mac OS and OS X only)
- `transparent` (Mac OS and OS X only)

If an object's ink property is one of the nine Mac OS-only transfer modes, it appears as though its ink were set to `srcCopy` when it's displayed on Unix and Windows systems.

The 15 modes that are not Mac OS-only (other than `blend`) should be used on systems whose `screenType` is `"TrueColor"` or `"DirectColor"`; that is, 16-bit or 24-bit. Using these transfer modes on screens with fewer colors may produce unexpected results.

Note: Whenever you set the `blendLevel` property of an image, its ink property is automatically set to `"blend"`.

insert script command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backScripts function, call command, frontScripts function, licensed function, remove script command, script property, start using command, How to call a custom function that's not in the message path, How to include an object in the message path of every object, How to intercept a message

Summary

Places an object's script into the message path.

Syntax

insert [the] script of object into {front | back}

Example Code

```
insert the script of button "Message Library" into back
insert the script of this card into front
```

Comments

Use the insert script command to use an object's script as a library for frequently-used handlers.

Parameters:

The object is any object in an open stack.

Comments:

A script inserted into the front receives messages first, before the target object receives them. A script inserted into the back receives messages after all objects in the message path, just before the engine itself receives the message.

Objects added to the front or back are placed at the start of the frontScripts or backScripts list: the last-inserted object gets messages first.

When using a standalone application, the number of frontScripts or backScripts that can be active at once is specified by the scriptLimits function, and is currently ten backScripts and ten frontScripts.

In the development environment, you can insert any number of scripts into the front or back of the message path. However, inserting scripts may make it difficult to understand how a stack behaves. If you use this command for development, be sure to carefully document which scripts are inserted, where, and when.

int1

keyword

Synonyms

int1s

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

int2 keyword, int4 keyword, read from file command, read from process command, read from socket command, uInt1 keyword

Summary

Used with the read from file, read from process, and read from socket commands to signify a chunk of binary data the size of a signed 1-byte integer.

Syntax

Example Code

```
read from process "serial.exe" for 1 int1
```

Comments

Use the int1 keyword to read data from a binary file.

Comments:

If you specify int1 as the chunk type in a read from file, read from process, or read from socket command, the data is returned as a series of numbers separated by commas, one for each signed integer read.

int2

keyword

Synonyms

int2s

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

int1 keyword, int4 keyword, read from file command, read from process command, read from socket command, uInt2 keyword

Summary

Used with the read from file, read from process, and read from socket commands to signify a chunk of binary data the size of a signed 2-byte integer.

Syntax

Example Code

```
read from process "trap" for 4 int2
```

Comments

Use the int2 keyword to read data from a binary file.

Comments:

If you specify int2 as the chunk type in a read from file, read from process, or read from socket command, the data is returned as a series of numbers separated by commas, one for each signed integer read.

int4

keyword

Synonyms

int4s

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

int1 keyword, int2 keyword, read from file command, read from process command, read from socket command, uInt4 keyword

Summary

Used with the read from file, read from process, and read from socket commands to signify a chunk of binary data the size of a signed 4-byte integer.

Syntax

Example Code

```
read from process "AppleScripts/Get Info" for 1 int4
```

Comments

Use the int4 keyword to read data from a binary file.

Comments:

If you specify int4 as the chunk type in a read from file, read from process, or read from socket command, the data is returned as a series of numbers separated by commas, one for each signed integer read.

integer
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

is a operator, is not a operator, Recipe for rounding a number up to a ceiling

Summary

Used with the is a and is not a operators to check whether a value is an integer.

Syntax

Example Code

```
if field "Number to Order" is not an integer then beep
```

Comments

Use the integer keyword to find out whether a value is a number that doesn't have a fractional part.

Comments:

If the value contains digits (and an optional minus sign) and no other characters, it is an integer.

If the value is an expression, it is evaluated and Revolution checks whether the final value is an integer. For example, the value of the expression "22 - 3" is 19 (which is an integer), so the following expression evaluates to true:

22 - 3 is an integer

However, the expression

"22 - 3" is an integer

evaluates to false, because the double quotes prevent the expression "22 - 3" from being evaluated.

internet

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

convert command, date function, english keyword, system keyword, time function

Summary

Specifies a format for the date function and the convert command.

Syntax

Example Code

```
put "Date:" && the internet date & return after field "Mail Header"
```

Comments

Use the internet keyword to work with dates in email headers and other dates generated by standard Internet protocols that use the same format.

Comments:

The internet form of a date consists of the following parts, separated by spaces:

- ī the day of the week followed by a comma
- ī the day of the month
- ī the three-letter abbreviation for the month name
- ī the four-digit year
- ī the time in 24-hour format, including seconds, delimited by colons
- ī the four-digit time zone relative to UTC (Greenwich) time

An internet date looks like this: Sun, 28 Oct 2001 17:18:54 -0800

You can use the modifiers long, short, and abbreviated in conjunction with the internet keyword without causing a script error, but they have no effect on the resulting date.

Note: The internet keyword is implemented internally as a property and appears in the propertyNames. However, it cannot be used as a property in an expression, nor with the set command.

For technical information about the date format produced by the internet keyword, see RFC 2822 at [<http://www.ietf.org/rfc/rfc2822.txt>](http://www.ietf.org/rfc/rfc2822.txt).

interrupt

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

allowInterrupts property, errorDialog message, lockErrorDialogs property, How to stop a running handler, Why can't I interrupt a handler?, Shortcut to stop a running handler

Summary

Returns true if the allowInterrupts property is false and the user has attempted to halt the current handler with a key combination.

Syntax

the interrupt

interrupt()

Example Code

```
if the interrupt then doCleanExit
```

Comments

Use the interrupt function within a handler to check whether the user has attempted to stop the handler by typing Control-period or Control-break (on Windows or Unix) or Command-period (on Mac OS).

Value:

The interrupt function returns true or false.

Comments:

Normally, pressing one of the key combinations stops any running handlers. If the allowInterrupts property is set to false, the user cannot interrupt a handler in this way.

To let the user interrupt a handler when the allowInterrupts is false, check the interrupt function periodically. If it returns true, the user has attempted to stop the handler. You can then perform any needed cleanup tasks before exiting the handler.

intersect

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

[] keyword, combine command, delete variable command, keys function, split command, union command, About containers, variables, and sources of value, How to find out whether a container is an array, Recipe for finding common lines in two containers

Summary

Removes elements from an array if they have no corresponding element in another array.

Syntax

intersect array with templateArray

Example Code

```
intersect myArray with the templateData of this stack
```

Comments

Use the intersect command to filter out elements from an array according to the contents of another array.

Parameters:

The array is any array variable.

The templateArray is any array variable.

Comments:

Each key of the array is checked to see whether there is a matching key in the templateArray. The elements of array that do not match an element of the templateArray are removed from the array.

The content of individual elements of the templateArray does not affect the final result. Only which elements exist in the templateArray, not their content, controls which elements of the array are retained and which are removed. If the array and templateArray have the same set of elements but different content in each element, the intersect command does not change the value of the array.

intersect

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

is within operator, layer property, within function

Summary

Returns true if two objects overlap, false otherwise.

Syntax

`intersect(object,object)`

Example Code

```
intersect(field "Comment",the selectedField)  
if intersect(button "target", button "crosshairs") then flashScreen
```

Comments

Use the intersect function to determine whether one object obscures part of another object, or whether one object is over another object.

Parameters:

The object is any object reference.

Value:

The intersect function returns true or false.

Comments:

If both objects are stacks or cards, the intersect function returns true if the stack windows overlap.

The intersect function uses screen coordinates for stacks and cards, and window coordinates for controls. This means that if one object is a stack or card and the other is a control, the intersect function does not return a reliable result.

intersect

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

select command, selection keyword, selectionMode property, surround keyword

Summary

Used with the selectionMode property to cause objects to be selected when any part of the object is within the rectangle you drag.

Syntax

Example Code

```
set the selectionMode to intersect
```

Comments

Use the intersect keyword to change the way selecting behaves.

Comments:

The intersect keyword applies only to selections made with the Pointer tool. It does not affect text selections in a field.

into
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

after keyword, before keyword, put command

Summary

Used with the put command to place a value in a container.

Syntax

Example Code

```
put myVariable into field "Contents"
```

Comments

Use the into keyword to designate a destination container.

Comments:

If the container is a variable name, and the variable does not already exist, the put command creates it as a local variable and sets it to the designated value.

If the container already exists, the into keyword replaces its former contents.

inverse
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

card keyword, gray keyword, hide command, inverse keyword, show command, visual effect command, white keyword

Summary

Used with the visual effect command, included in Transcript for compatibility with imported HyperCard stacks.

Syntax

Example Code

```
visual effect dissolve to inverse
```

Comments

In HyperCard, the inverse keyword indicates that the color inverse of the current card is shown at the end of the visual effect.

In Revolution, the inverse keyword is a synonym of the white keyword.

invisible

property

Synonyms
inv

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
disable command, enable command, hide command, lockScreen property, show command, showInvisibles property, visible property, How to peek at invisible objects in a stack

Summary
Specifies whether an object is hidden.

Syntax
set the invisible of object to {true | false}

Example Code
set the invisible of player "Splash Screen" to false
set the invisible of the mouseControl to true

Comments
Use the invisible property to determine whether an object is hidden or not.

Value:
The invisible of an object is true or false.

Comments:
A hidden object is still present and still takes up memory, and a handler can access its properties and contents, but the user cannot see or or interact with it.

An object that cannot be seen only because it's behind another object is still visible, in the sense that its invisible property is still false.

The invisible property of grouped controls is independent of the invisible property of the group. Setting a group's invisible property to true doesn't change the invisible property of its controls; their invisible property is still false, even though the controls cannot be seen because the group is invisible.

You can set the invisible property of a card, but doing so has no effect. Cards cannot be made invisible.

The invisible property is the logical inverse of the visible property. When an object's invisible is true, its visible is false, and vice versa.

is a
operator

Synonyms
is an

Objects
logical

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
colorNames function, convert command, date function, false constant, integer keyword, is not a operator, numberFormat property, point keyword, there is a operator, time function, true constant, Color Names Reference, Operator Precedence Reference, Recipe for rounding a number up to a ceiling, Recipe for translating a color name to an RGB numeric triplet

Summary
Evaluates to true if a value is of the specified type, false otherwise.

Syntax
value is a[n] {boolean | integer | number | point | rect | date | color}

Example Code
"1/16/98" is a date -- evaluates to true
1 is a boolean -- evaluates to false
45.4 is an integer -- evaluates to false
"red" is a color -- evaluates to true

Comments
Use the is a operator to validate data to make sure it's the right type.

Parameters:
The value is any source of value.

Comments:
This operator is useful for checking whether the user has entered data correctly, and for checking parameters before sending them to a handler to avoid a script error caused by feeding data of one type to an operator or function that requires a different type.

A value is a:

- boolean or logical if it is one of the two constants true or false

- integer if it consists of digits (with an optional leading minus sign)

• number if it consists of digits, optional leading minus sign, optional decimal point, and optional "E" or "e" (scientific notation)

• point if it consists of two numbers separated by a comma

• rect if it consists of four numbers separated by commas

• date if it is in one of the formats produced by the date or time functions

• color if it is a valid color reference

All the types other than boolean can also include leading or trailing white space characters.

The `is a` operator is the logical inverse of the `is not a` operator. When one is true, the other is false.

is among operator

Synonyms

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

contains operator, is in operator, is not among operator, itemOffset function, lineOffset function, wholeMatches property, wordOffset function, Operator Precedence Reference

Summary

Evaluates to true if a chunk is equal to one of the chunks in a string, false otherwise.

Syntax

valueToFind is among the {words | items | lines} of stringToSearch

Example Code

```
"Hello" is among the words of "Hello World" -- evaluates to true
```

```
"Hell" is among the words of "Hello World" -- evaluates to false
```

Comments

Use the is among operator to find out whether a string exists as an entire word, item, or line of another string.

Parameters:

The valueToFind is a string or an expression that evaluates to a string.

The stringToSearch is a string is an expression that evaluates to a string.

Comments:

The valueToFind must be a single word, item, or line (whichever chunk type you have specified). If you specify a string containing more than one chunk, the is among operator evaluates to false, even if all the chunks are in the stringToSearch.

The is among operator is the logical inverse of the is not among operator. When one is true, the other is false.

is in
operator

Synonyms

Objects
logical

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

contains operator, is not in operator, Operator Precedence Reference

Summary

Compares two strings and evaluates to true if the second contains the first, false if not.

Syntax

subString is in string

Example Code

```
"A" is in "ABC" -- evaluates to true  
"123" is in "13" -- evaluates to false
```

Comments

Use the is in operator to find out whether a string contains a substring.

Parameters:

The substring and string are both strings, or expressions that evaluate to strings.

Comments:

The expression
firstString is in secondString
is equivalent to
secondString contains firstString

The is in operator is the logical inverse of the is not in operator. When one is true, the other is false.

is not a
operator

Synonyms
is not an

Objects
logical

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
convert command, date function, integer keyword, is a operator, point keyword, there is no operator, time function, Color Names Reference, Operator Precedence Reference, Recipe for translating a color name to an RGB numeric triplet

Summary
Evaluates to true if a value is not of the specified type, false otherwise.

Syntax
value is not a[n] {boolean | integer | number | point | rect| date | color}

Example Code
`22 is not a number -- evaluates to false`
`CJ22 is not a number -- evaluates to true`

Comments
Use the is not a operator to validate data to make sure it's not the wrong type.

Parameters:
The value is any source of value.

Comments:
This operator is useful for checking whether the user has entered data correctly, and for checking parameters before sending them to a handler to avoid a script error caused by feeding data of one type to an operator or function that requires a different type.

A value is a:

- boolean or logical if it is one of the two constants true or false

- integer if it consists of digits (with an optional minus sign)

ĩ number if it consists of digits, optional leading minus sign, optional decimal point, and optional "E" or "e" (scientific notation)

ĩ point if it consists of two numbers separated by a comma

ĩ rect if it consists of four numbers separated by commas

ĩ date if it is in one of the formats produced by the date or time functions

ĩ color if it is a valid color reference

All the types other than boolean can also include leading or trailing white space characters.

The is not a operator is the logical inverse of the is a operator. When one is true, the other is false.

is not among operator

Synonyms

Objects logical

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

contains operator, is among operator, is not in operator, itemOffset function, lineOffset function, not operator, wholeMatches property, wordOffset function, Operator Precedence Reference

Summary

Evaluates to true if a chunk is not equal to any of the chunks in a string, false otherwise.

Syntax

valueToFind is not among the {words | items | lines} of stringToSearch

Example Code

```
"a" is not among the words of "human" -- evaluates to true
```

Comments

Use the is not among operator to find out whether a string exists as an entire word, item, or line of another string.

Parameters:

The valueToFind is a string or an expression that evaluates to a string.

The stringToSearch is a string is an expression that evaluates to a string.

Comments:

The valueToFind must be a single word, item, or line (whichever chunk type you have specified). If you specify a string containing more than one chunk, the is not among operator evaluates to true, even if all the chunks are in the stringToSearch.

The is not among operator is the logical inverse of the is among operator. When one is true, the other is false.

is not in
operator

Synonyms

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

contains operator, is in operator, not operator, Operator Precedence Reference

Summary

Compares two strings and evaluates to true if the second does not contain the first, false if it does.

Syntax

subString is not in string

Example Code

```
"S" is not in "BSD" -- evaluates to false
```

Comments

Use the is not in operator to find out whether a string contains a substring.

Parameters:

The substring and string are both strings, or expressions that evaluate to strings.

Comments:

The expression

firstString is not in secondString

is equivalent to

not (secondString contains firstString)

The is not in operator is the logical inverse of the is in operator. When one is true, the other is false.

is not within
operator

Synonyms

Objects
logical

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
is within operator, within function, Operator Precedence Reference

Summary
Evaluates to true if a point is outside the specified rectangle, false if it is inside.

Syntax
point is not within rectangle

Example Code
`"0,0" is not within "300,300,350,350" -- evaluates to true`

Comments
Use the is not within operator to determine whether a point is inside a rectangle.

Parameters:
The point is any expression that evaluates to a pointó a vertical and horizontal distance from the top left of the current stack, separated by a comma.

The rectangle is any expression that evaluates to a rectangleó a right, top, left, and bottom edge, separated by commas.

Comments:
The is not within operator is the logical inverse of the is within operator. When one is true, the other is false.

is within operator

Synonyms

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

is not within operator, within function, Operator Precedence Reference

Summary

Evaluates to true if a point is inside the specified rectangle, false if it is outside.

Syntax

point is within rectangle

Example Code

```
"22,33" is within "22,17,150,200" -- evaluates to true  
the mouseLoc is within the rect of this stack
```

Comments

Use the is within operator to determine whether a point is inside a rectangle.

Parameters:

The point is any expression that evaluates to a point—a vertical and horizontal distance from the top left of the current stack, separated by a comma.

The rectangle is any expression that evaluates to a rectangle—a right, top, left, and bottom edge, separated by commas.

Comments:

The expression

point is within the rect of object

is equivalent to

within(object,point)

unless the object is a graphic or image, or a tabbed button.

The is within operator is the logical inverse of the is not within operator. When one is true, the other is false.

isNumber

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

exists function, is a operator

Summary

Returns true if a value is a number, false if it is not.

Syntax

the isNumber of value

isNumber(value)

Example Code

```
isNumber(8) -- returns true  
isNumber(1+5) -- returns true  
isNumber(foo) -- returns false
```

Comments

Use the isNumber function to determine whether a value is a number. For example, attempting to do an arithmetic operation on a value that's not a number causes an error. You can check the value beforehand to prevent this.

Parameters:

The value is any source of value.

Value:

The isNumber function returns true or false.

Comments:

This function exists for compatibility with imported SuperCard projects. You can use the is a operator to validate numbers, as well as other data types.

ISOToMac

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

charSet property, charToNum function, macToISO function, numToChar function, platform function

Summary

Returns the equivalent of an ISO 8859-1 string, in the Mac OS character set.

Syntax

the ISOToMac of ISOString

ISOToMac(ISOString)

Example Code

```
ISOToMac("Ab4+") -- returns "Ab4+" since there are no special chars  
ISOToMac("...") -- returns È, the Mac equivalent  
charToNum( ISOToMac(numToChar(myASCIIVal))) -- returns ASCII equivalent
```

Comments

Use the ISOToMac function to translate data that was created on a Unix or Windows system to the Mac OS character set.

Parameters:

The ISOString is any string.

Value:

The ISOToMac function returns the ISOString, with characters whose ASCII value is greater than 127 converted to their equivalent in the Mac OS character set. Characters whose ASCII value is less than 128 are left unchanged.

Comments:

Revolution automatically translates text in fields and scripts, as well as the names of custom properties, into the appropriate character set when you move a stack from one platform to another. It is therefore not necessary to translate them. However, the contents of custom properties, since they may contain binary data, are not translated automatically and must be translated if they contain characters whose ASCII value is 128 or greater.

Characters whose ASCII value is less than 128 are the same in the Mac OS character set and ISO 8859-1, so they are not changed by the ISOToMac function. These characters include uppercase and lowercase letters, numbers, and most punctuation.

it
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

answer command, answer file command, answer folder command, ask command, ask file command, convert command, get command, read from file command, read from process command, variableNames function

Summary

A special local variable that is used with commands such as get, read from file, convert, ask, and answer.

Syntax

Example Code

```
get the date  
put it into field "Date"
```

Comments

Use the it keyword to get the result of certain commands, or as a handy temporary storage place.

Comments:

The it keyword designates a special local variable. It can be used like any other local variable: you can put a value into it, or put it into another container.

The commands that use the it variable are: answer, answer color, answer effect, answer file, answer folder, ask, ask file, ask password, clone, convert, copy, create, get, paste, post, read from driver, read from file, read from process, read from socket, request, and request appleEvent.

Unlike other local variables, it can change even if you don't explicitly change its value. Certain commands use it to store their result. Therefore, if you need to use the value of it, make sure none of these commands is executed between the time you set it and the time you read its value. If you need to save the value of it for later use, store it in another variable:

```
put it into savedFilePath
```

italic

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bold keyword, plain keyword, strikeout keyword, textStyle property, underline keyword, Text menu > Italic

Summary

Used with the textStyle property to indicate italicized text.

Syntax

Example Code

```
set the textStyle of button "Check It" to italic
```

Comments

Use the italic keyword to emphasize text by presenting it in italics.

Comments:

You can italicize an object (which italicizes all the text displayed by the object) or a chunk in a field or button.

To add italics to an object without removing other styles (such as bold), add the italic keyword to the end of the textStyle. The following example adds italic to the styles of a button:

```
if the textStyle of button 1 is empty then
    set the textStyle of button 1 to "italic"
else
    set the textStyle of button 1 to
```

```
    (the textStyle of button 1) & comma & "italic"
end if
```


item

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

character keyword, itemDelimiter property, line keyword, token keyword, word keyword, About chunk expressions

Summary

Designates a comma-delimited string as part of a chunk expression.

Syntax

Example Code

```
get item 2 of field "Name"  
put it into item 3 of line thisRecord of myNamesDB
```

Comments

Use the item keyword to refer to a specific chunk of data in a container.

Comments:

By default, an item is delimited by commas (.). However, you can set the itemDelimiter property to any character in order to directly address chunks delimited by that character. For example, you can set the itemDelimiter to tab in order to handle a database file whose fields are tab-delimited.

An item is a string of characters delimited by a single comma (or a single one of the itemDelimiter character). A single item can contain multiple characters, multiple words, and multiple lines.

itemDelimiter

property

Synonyms

itemDel

Objects

local

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

colon constant, comma constant, formfeed constant, item keyword, itemOffset function, items keyword, lineDelimiter property, return constant, Recipe for converting an absolute path to a relative path, Recipe for converting a relative path to an absolute path

Summary

Specifies the character used to separate items in chunk expressions.

Syntax

set the itemDelimiter to character

Example Code

```
set the itemDelimiter to numToChar(202)
set the itemDelimiter to tab
```

Comments

Use the itemDelimiter property to divide text into chunks based on a delimiting character.

Value:

The itemDelimiter is a character.

By default, the itemDelimiter property is set to comma.

Comments:

The itemDelimiter is a single character. Chunk expressions use the itemDelimiter to determine where one item ends and the next begins.

Since the itemDelimiter is a local property, its value is reset to comma when the current handler finishes executing. It retains its value only for the current handler, and setting it in one handler does not affect its value in other handlers it calls.

itemOffset

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

caseSensitive property, is among operator, item keyword, itemDelimiter property, items keyword, lineOffset function, offset function, wholeMatches property, wordOffset function

Summary

Returns the number of items between the beginning of a value and an occurrence of a specified string.

Syntax

`itemOffset(itemToFind,stringToSearch[,itemsToSkip])`

Example Code

```
itemOffset("C","A,B,C,D,E") -- returns 3
itemOffset("C","A,B,C,D,E",2) -- returns 1
itemOffset("D,E","A,B,C,D,E") -- returns 4
```

Comments

Use the itemOffset function to find which item a string occurs in.

Parameters:

The itemToFind is a string or an expression that evaluates to a string.

The stringToSearch is a string or an expression that evaluates to a string.

The itemsToSkip is a non-negative integer. If you don't specify how many itemsToSkip, the itemOffset function does not skip any items.

Value:

The itemOffset function returns a non-negative integer.

Comments:

The value returned by the itemOffset function is the number of the item where itemToFind appears in stringToSearch. If the itemToFind is not in stringToSearch, the itemOffset function returns zero.

If the `itemToFind` contains more than one item, and the entire `itemToFind` appears in the `stringToSearch`, the `itemOffset` function returns the item number where the `itemToFind` starts.

If you specify how many `itemsToSkip`, the `itemOffset` function skips the specified number of items in the `stringToSearch`. The value returned is relative to this starting point instead of the beginning of the `stringToSearch`.

items

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

characters keyword, is among operator, is not among operator, item keyword, itemDelimiter property, lines keyword, number function, sort command, words keyword

Summary

Used with the sort command, number function, and is among and is not among properties to designate the comma-delimited portions (or portions delimited by the character in the itemDelimiter property) of a string.

Syntax

Example Code

```
if it is not among the items of field "Possible Answers" then beep
sort items of myVariable
```

Comments

Use the items keyword to sort or select individual items.

Comments:

By default, an item is delimited by commas (.). However, you can set the itemDelimiter property to any character in order to directly address chunks delimited by that character. For example, you can set the itemDelimiter to tab in order to handle a database file whose fields are tab-delimited.

An item is a string of characters delimited by commas (or the itemDelimiter character). A single item can contain multiple characters, multiple words, and multiple lines.

JPEGQuality

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

export command, import command, paintCompression property

Summary

Specifies the quality level of JPEG images created by Revolution.

Syntax

set the jpegQuality to qualityNumber

Example Code

```
set the JPEGQuality to 40  
set the JPEGQuality to the JPEGQuality - 10
```

Comments

Use the JPEGQuality property to control the file size and sharpness of JPEG files created with the export command.

Value:

The JPEGQuality is an integer between 1 and 100.

By default, the JPEGQuality property is set to 100.

Comments:

JPEG compression is inherently lossy, which means some detail is always lost when an image is compressed using this method. The JPEGQuality property controls how much detail is lost. If the JPEGQuality is 100, as much detail as possible is retained.

In general, the lower the quality setting, the smaller the file size. Photographic images with naturally fuzzy edges generally do not suffer much visible loss of quality with a JPEGQuality setting of 50 or higher. Different images may require different levels of this property to achieve acceptable results.

The JPEGQuality setting is used when a file is exported in JPEG format using the export command, and when an image whose paintCompression property is "jpeg" is changed.

keyDown

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

capsLockKey function, commandKey function, enterKey message, escapeKey message, keysDown function, keyUp message, mouseDown message, optionKey function, rawKeyDown message, returnKey message, shiftKey function, type command, Recipe for limiting the number of characters in a field, Recipe for rejecting non-digit characters typed into a field, How to prevent entering certain characters in a field

Summary

Sent when the user presses a key.

Syntax

keyDown keyName

Example Code

```
on keyDown theKey -- block typing anything but digits
  if theKey is not a number then beep
  else pass keyDown
end keyDown
```

Comments

Handle the keyDown message if you want to do something special when the user presses any key or a particular key you check for in the handler.

Parameters:

The keyName is the actual character of the pressed key.

Comments:

The message is sent to the active (focused) control, or to the current card if no control is focused.

If the key pressed is the Return, Tab, Backspace, Delete, or Enter key, an arrow key, or a function key, no keyDown message is sent. Instead, the returnKey, tabKey, backspaceKey, deleteKey, enterKey, arrowKey, or functionKey message is sent.

If the insertion point is in a field, the entry of typed characters is triggered by the keyDown message. This means that trapping the keyDown message and not passing it prevents typing from being entered in the field.

Note: The keyDown message is sent before the character is placed in the field, so if you need to use the new content of the field, handle the keyUp message instead.

keys

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

[] keyword, combine command, customKeys property, delete variable command, properties property, split command, variableNames function, About containers, variables, and sources of value, How to find out the number of elements in an array, How to find out whether a container is an array, Recipe for finding common lines in two containers, Recipe for listing the unique words in a string

Summary

Returns a list of the element names in an array variable.

Syntax

the keys of arrayName
keys(arrayName)

Example Code

```
the keys of myArray  
repeat with thisItem = 1 to the number of lines in the keys of it
```

Comments

Use the keys function to manage the elements of an array, or to perform some action on each element in an array.

Parameters:

The arrayName is the name of a variable.

Value:

The keys function returns a list of keys, one per line.

Note: The order of the keys is not alphabetical or chronological; it is based on the internal order. To obtain an alphabetical list of keys, use the sort command:

```
put the keys of myArray into myVariable  
sort lines of myVariable
```

keysDown

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

altKey function, capsLockKey function, commandKey function, controlKey function, keyDown message, keyUp message, metaKey function, optionKey function, rawKeyDown message, rawKeyUp message, shiftKey function

Summary

Returns a list of the keys currently being pressed.

Syntax

the keysDown
keysDown()

Example Code

```
the keysDown  
if myKey is in the keysDown then exit to top
```

Comments

Use the keysDown function to respond to a keypress while a handler is being executed.

Value:

The keysDown function returns a list of keycodes of pressed keys, separated by commas if more than one key is being pressed.

Comments:

The keycodes returned by the keysDown function are the same as those passed as parameters with the rawKeyDown and rawKeyUp messages.

The keysDown function returns empty if no keys are being pressed.

keyUp

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

capsLockKey function, commandKey function, keyDown message, keysDown function, mouseUp message, optionKey function, rawKeyUp message, shiftKey function, type command

Summary

Sent when the user releases a pressed key.

Syntax

keyUp keyname

Example Code

```
on keyUp -- play a keyclick sound
  -- in this example, the parameter is not needed or used
  play audioClip "click"
end keyUp
```

Comments

Handle the keyUp message if you want to do something special when the user releases any key (or a particular key you check for in the handler).

Parameters:

The keyName is the actual character of the pressed key.

Comments:

The message is sent to the active (focused) control, or to the current card if no control is focused.

If the key pressed is the Return, Tab, Backspace, Delete, or Enter key, an arrow key, or a function key, no keyUp message is sent.

Cross-platform note: On Mac OS systems, the keyUp message is sent after any keyDown handlers finish executing, whether or not the key has been released. On Unix and Windows systems, the keyUp message is sent when the key is released, after the typed character has been added to the current field. To test whether a key is actually being pressed, use the keysDown function.

kill

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close process command, open process command, openProcesses function, openProcessIDs function, processID function, signal message

Summary

Signals or quits another process or application on the same system.

Syntax

kill [signalNumber | signalName] process processName

Example Code

```
kill process "rnews"  
kill 9 process myProcess -- terminate with extreme prejudice  
kill QUIT process it
```

Comments

Use the kill command to send a signal to a process (on Unix systems), or to terminate a process with extreme prejudice.

Parameters:

The signalNumber is the number of the Unix signal to send to the process. The signalName is the name of a Unix signal, minus the leading "SIG". (For example, to send SIGHUP to a process, use HUP as the signalName.) (The signalNumber or signalName parameter is ignored on Mac OS and Windows systems.)

The processName is the name of a currently executing process.

Comments:

On Mac OS systems, the kill command sends a "Quit Application" Apple event to the specified application.

On Unix systems, the kill command sends the specified signal to the process. If no signal is specified, the kill command sends SIGTERM. Check the Unix documentation for information about available signals. The file `/usr/include/sys/signal.h` lists signals and their corresponding signal numbers.

If possible, use the close process command instead of the kill command to terminate a process. The kill command causes an immediate exit, and may prevent the process from removing temporary files or doing other cleanup tasks.

label

property

Synonyms

title

Objects

button, graphic, group, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

effective keyword, fileName property, menuHistory property, showName property, How to change the selected item in an option menu or combo box, How to display the same text on multiple cards, How to remove a window's title bar, How to switch between button tabs

Summary

Specifies the string shown in a stack window's title bar, or a text label to be displayed on the specified object if its showName property is true.

Syntax

set the label of object to labelString

Example Code

```
set the label of button 1 to "Hello" & return & "World"
get the label of button "Choices Popup" -- current menu choice
set the label of this stack to the short name of this card
```

Comments

Use the label property as a user-visible replacement for an ungainly name, or to change the visible name of a window or object when changing the actual name would require changing code that refers to it.

Value:

The label of an object is a string.

By default, the label property of newly created buttons, graphics, groups, and stacks is set to empty.

Comments:

If a button's menuMode is either "option" or "comboBox", the button's label is the text of the currently selected menu option. Setting the label property changes the currently selected option. (To change the currently selected option while sending the appropriate message, set the button's menuHistory instead.)

To create a multiple-line label for a button or graphic, place a return constant in the label.

If an object's label is empty, the object's name property is displayed instead. In this case, the expression the effective label of button reports the button's name property.

To create a blank title bar, set the stack's label property to a space.

last

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

any keyword, first keyword, layer property, middle keyword, number property

Summary

Specifies the last member of a set of objects or the last chunk of a string.

Syntax

Example Code

```
put the ID of last card into stoppingPoint  
put the last line of it after field "Answers"
```

Comments

The last keyword can be used to specify any object whose number property is equal to the number of objects of that type. It can also be used to designate the last chunk in a chunk expression.

The last control is the topmost control on a card. The last button is the topmost button, and similarly for other types of controls.

The word the is optional when using the last keyword.

A reference to the last of a string, with no chunk type specified, yields the entire string.

launch

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

kill command, open process command, openProcesses function, openProcessIDs function, revGoURL command, secureMode property, shell function

Summary

Starts up an application, optionally opening a document in the application.

Syntax

launch [documentPath with] applicationPath

Example Code

```
launch "SimpleText"  
launch "/Documents/Projects/test.txt" with myApp  
launch it with (field "Application")
```

Comments

Use the launch command to start an application for the user to use.

Parameters:

The documentPath is the location and name of a file to open with the specified application. If no path is specified, the launch command assumes that the file is in the defaultFolder.

The applicationPath is the location and name of the application to start up. If no path is specified, the launch command assumes that the application is in the defaultFolder.

Comments:

When the launch command executes, the application being launched comes to the foreground. When the user quits, Revolution comes to the foreground.

If the application is already running, the launch command does nothing, and "Process is already open." is placed in the result function.

If no documentPath is specified, the following two statements are equivalent:

launch application
open process application for neither

Note: On OS X systems, you can use the launch command to start up an application, but not a Unix process. To work with a Unix process, use the shell function instead.

Tip: On Windows systems, you can also start up an application by using the shell function with the Windows "start" command:

```
get shell("start MyProgram.exe")
```

Changes to Transcript:

In versions before 1.1.1, when you quit the application, any applications that had been launched with the launch command were quit automatically on Windows systems.

layer

property

Synonyms
partNumber

Objects
control, card

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
first keyword, group command, last keyword, number property, relayerGroupedControls property, ungroup command, How to bring a control to the front, How to change the order of cards in a stack, How to keep a window on top of other windows, Object menu > Send to Back, Object menu > Move Backward, Object menu > Move Forward, Object menu > Bring to Front, Window menu > Send Window to Back

Summary
Specifies the back-to-front order of objects on a card or the order of cards in a stack.

Syntax
set the layer of object to {layerNumber | top | bottom}

Example Code

```
set the layer of the mouseControl to top -- bring to front
set the layer of last card button to 10
set the layer of player "Splash" to (the number of controls)
```

Comments
Use the layer property to change the tab order of controls, or to change the order of cards within a stack.

Value:
The layer property of a control or card is an integer.

Comments:
If the object is a control, it must appear on the current card in order for its layer to be changed; you can't set the layer of a control on another card.

The layer of a control is its order (from back to front) on the card. If two controls overlap, the one that covers the other has a higher layer. Each control is on a different layer.

To bring a control forward, set its layer to a higher number. To send it back, set its layer to a lower number. You use the form

set the layer of object to top
to bring the control all the way to the front, and
set the layer of object to bottom
to send it all the way to the back.

If you set the layer of a control to a number greater than the number of controls on the card, its layer is set equal to the number of controls.

The layer of controls also determines the tab order. When you press the Tab key, the next control whose `traversalOn` property is true becomes the active (focused) control. For example, to set the tab order of fields in a data entry form, set the layer of each field so that they are in order.

The layer of a group determines the back-to-front position of all the controls in the group at once. The group behaves as a single control, as far as its layer is concerned. When you group or ungroup a group, its controls retain the same back-to-front placement relative to each other, but the group as a whole moves to the front. The layer of a group is one less than the layer of the lowest-layered object in the group.

It is not possible for other controls to be interspersed between the controls in a single group. If the `relayerGroupedControls` property is set to true, and you set the layer of a grouped control to a number greater than the topmost control in the group, or lower than the bottom-most control in the group, the control is moved out of the group. If the `relayerGroupedControls` is false, you cannot set the layer of a grouped control without being in group-editing mode.

The layer of a card specifies its position in the stack. Setting the layer of a card moves it to the specified position in the stack.

Changing the layer of an object also changes its number property.

left

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

altKey function, commandKey function, controlKey function, down constant, left constant, mouse function, optionKey function, right constant, shiftKey function

Summary

Equivalent to the string "left".

Syntax

Example Code

```
arrowKey left  
if theKey is left then go previous card
```

Comments

Use the left constant to indicate the left arrow key in an arrowKey handler.

left

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

bottomLeft property, leftMargin property, right property, topLeft property, width property, Object menu > Align Selected Controls > Left, Shortcut to align control edges

Summary

Specifies how far an object's left edge is from the left edge of the window or screen.

Syntax

set the left of object to numberOfPixels

Example Code

```
set the left of graphic 3 to the left of graphic 2
set the left of this stack to 128
```

Comments

Use the left property to change the horizontal placement of a control or window.

Value:

The left of an object is an integer. A negative integer indicates that the position is to the left of the left edge of the screen or card (and therefore cannot be seen).

A stack's left is the distance in pixels from the left edge of the screen to the left edge of the stack window.

A card's or control's left is the distance in pixels from the left edge of the card to the left edge of the card or control.

Comments:

The left of a stack is in absolute (screen) coordinates. The left of a card is always zero; setting the left of a card does not cause a script error, but it has no effect. The left of a group or control is in relative (window) coordinates.

Changing the left of an object shifts it to the new position without resizing it. To change an object's width, set its width or rectangle property.

The width property of an object is equal to its right minus its left.

leftMargin

property

Synonyms

Objects

button, field, group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomMargin property, firstIndent property, formattedWidth property, left property, margins property, rightMargin property, textAlign property, topMargin property, width property

Summary

Specifies how close text within an object can come to the object's left edge, and how close objects in a group can come to the group's left edge.

Syntax

set the leftMargin of {button | field | group} to pixels

Example Code

```
set the leftMargin of field "Melting Point" to 10
set the leftMargin of last button to the leftMargin of first button
```

Comments

Use the leftMargin property to change the amount of blank space between an object's left edge and its contents.

Value:

The leftMargin is a non-negative integer.

By default, the leftMargin of a newly created field is 8. If the field's wideMargins property is true, the field's leftMargin is set to 14. The default leftMargin setting for a button or group is 4.

Comments:

The leftMargin property of a field or button specifies how many blank pixels are left between the object's left edge and the left edge of its text. The leftMargin of a group specifies how far the group's left edge extends below its leftmost object.

An object's leftMargin property is equal to item 1 of its margins property.

length

function

Synonyms
len

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

number property, offset function, Recipe for limiting the number of characters in a field, Recipe for word-wrapping text to a line length

Summary

Returns the number of characters in a string.

Syntax

the length of string
length(string)

Example Code

```
the length of "ABC" -- returns 3  
repeat with x = 1 to the length of receivedData  
  put char (length(it) - 3) to (length(it)) of it into myExtension
```

Comments

Use the length function to find out how many characters are in a string, in order to process each character.

Parameters:

The string is any string, or an expression that evaluates to a string.

Value:

The length function returns a non-negative integer.

Comments:

The length of a string is the number of characters in the string. The expression
 the length of string
is equivalent to
 the number of chars in string

If the string is empty, its length is zero.

libraryStack

message

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

releaseStack message, start using command, stop using command, How to create a code library

Summary

Sent to a stack when it is placed in the message path by the start using command.

Syntax

libraryStack

Example Code

```
on libraryStack -- show visually in the stack that it's in use
  set the hilite of button "Using" of card 1 of this stack to true
end libraryStack
```

Comments

Handle the libraryStack message if you want to perform some task or set a configuration when a stack is placed in use.

Comments:

The libraryStack message is sent to a stack even if it was already in use before the start using command was executed.

libURLDownloadToFile

command

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

ftp keyword, libURLftpUploadFile command, libURLSetFTPStopTime command, libURLSetStatusCallback command, load command, put command, unload command, URLStatus function, How to cancel a file transfer in progress, How to fetch data from the Internet, Why was a downloaded file corrupted?

Summary

Downloads a file from an Internet server asynchronously via FTP.

Syntax

libURLDownloadToFile downloadURL,filePath[,callbackMessage]

Example Code

```
libURLDownloadToFile "ftp://example.org/new_beta",it  
libURLDownloadToFile myFile,newPath,"downloadComplete"
```

Comments

Use the libURLDownloadToFile command to download a file from an FTP server.

Parameters:

The downloadURL specifies the server and file to download, in the form of an ftp URL.

The filePath specifies the name and location to save the downloaded file in. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

The callbackMessage is the name of a message to send after the download is finished.

Comments:

The libURLDownloadToFile command transfers the data directly from the file to the server. Unlike simply using the put command to put the contents of an ftp URL into a file, using the libURLDownloadToFile command does not load all the data into memory at once, so this command is a better choice for large files.

The `libURLDownloadToFile` command transfers the file in binary mode.

The `callbackMessage` is sent to the object whose script contains the `libURLDownloadToFile` command, after the download is complete, so you can handle the `callbackMessage` to perform any tasks you want to delay until the file has been downloaded. Two parameters are sent with the message: the URL and the `URLStatus` of the file.

Tip: On Mac OS and OS X systems, the new file's creator signature and file type is set to the value specified in the `fileType` property.

Important! The `libURLDownloadToFile` command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `libURLDownloadToFile` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `libURLDownloadToFile` command can be used in any handler.

libURLErrorData

function

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cachedURLs function, ftp keyword, http keyword, libURLSetLogField command, load command, URLStatus function

Summary

Returns any error that was caused during a download that was started with the load command.

Syntax

libURLErrorData(url)

Example Code

```
put libURLErrorData("http://www.example.org/index.html") into myErr
if libURLErrorData(it) contains "404" then answer "File doesn't exist!"
if libURLStatus(myURL) is "error" then return libURLErrorData(myURL)
```

Comments

Use the libURLErrorData function to check the error status of a download.

Parameters:

The url is a URL, or an expression that evaluates to a URL.

Value:

The libURLErrorData function returns a string.

Comments:

You can check the URLStatus function to determine the status of a URL that has been cached with the load command. If the download encountered an error, the URLStatus function returns "error". In this case, the libURLErrorData function returns the result of the attempted download.

If the download failed because of a server error—for example, if a requested web page was not found—the libURLErrorData function returns the error message sent by the server. If the download failed for another reason—for example, if the server did not respond, or if an invalid URL was supplied—the libURLErrorData function returns an error message generated by the Internet library.

Important! The `libURLErrorData` function is part of the Internet library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

libURLftpCommand

function

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

ftp keyword, libURLSetFTPListCommand command

Summary

Sends an FTP command to an FTP server.

Syntax

`libURLftpCommand(ftpCommandLine,host[:port][,username[,password]])`

Example Code

```
put libURLftpCommand("PWD",ftp.example.org) into field "Directory"  
get libURLftpCommand("SITE HELP",example.net:75,"root",field "password")
```

Comments

Use the libURLftpCommand function to communicate with an FTP server and perform tasks that are not implemented in the Internet library.

Parameters:

The ftpCommandLine is a string consisting of any valid FTP command and any parameters it requires.

The host is the IP address or domain name of the FTP server.

The port is the port number you want to connect to. If you don't specify a port, port 21 is used. (On most systems, port 21 is the FTP port.)

The username is the account name on the server. If you don't specify a username or password, the libURLftpCommand function uses the "anonymous" user name and a dummy password automatically, in accordance with the conventions for public FTP servers.

The password is the password to the username account.

Value:

The `libURLftpCommand` function returns a string with the result of the FTP command (if the server sends back a result) or an error message. An FTP server always begins its response with a three-digit code, followed by any other relevant information.

Comments:

The ability to send any FTP command to an FTP server is a powerful function that can enable you to provide full support for all FTP functionality in your application. To use it effectively, you must be familiar with FTP commands and responses and be able to properly construct an FTP command line.

Tip: To get general information about an FTP server, issue the FTP "HELP" command:

```
answer libURLftpCommand("HELP",ftp.example.net)
```

For technical information about FTP server commands, see RFC 959 at <http://www.ietf.org/rfc/rfc959.txt>, particularly section 4.1.3.

Important! The `libURLftpCommand` function is part of the Internet library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `libURLftpCommand` function is not yet available. This may affect attempts to use this function in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `libURLftpCommand` function can be used in any handler.

libURLftpUpload

command

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

ftp keyword, libURLftpUploadFile command, libURLSetFTPStopTime command, libURLSetStatusCallback command, load command, put command, unload command, URLStatus function, How to create a directory on an FTP server

Summary

Uploads data to an Internet server asynchronously via FTP.

Syntax

libURLftpUpload value,uploadURL[,callbackMessage]

Example Code

```
libURLftpUpload field "Data","ftp://ftp.example.org/file.txt"  
libURLftpUpload URL "binfile:data.jef",myURL,"uploadDone"  
libURLftpUpload myData,"ftp://me:secret@example.net/file.txt"
```

Comments

Use the libURLftpUpload command to put a file on a server.

Parameters:

The value is any expression that evaluates to a string.

The uploadURL specifies the server and location to upload to, in the form of an ftp URL.

The callbackMessage is the name of a message to send after the URL is uploaded.

Comments:

The libURLftpUpload command is non-blocking, so it does not stop the current handler while the upload is completed. The handler continues while the libURLftpUpload command uploads the URL in the background. You can monitor the upload by checking the URLStatus function periodically.

To upload a URL while blocking other operations, use the put command instead.

To upload a file, use the file URL type (for text files) or the binfile URL type (for binary files). Because referring to a file URL's contents loads the file into memory, if you are uploading a large file, make sure you have enough memory available. You can also use the `libURLftpUploadFile` command to upload a file.

The `callbackMessage` is sent to the object whose script contains the `libURLftpUpload` command, after the upload is complete, so you can handle the `callbackMessage` to perform any tasks you want to delay until the URL has been uploaded. Two parameters are sent with the message: the URL and the `URLStatus` of the file.

Caution! Avoid using the `wait` command in a handler after executing the `libURLftpUpload` command. Since the `libURLftpUpload` command is non-blocking, it may still be running when your handler reaches the `wait` command. And since the `libURLftpUpload` command is part of the Internet library and is implemented in a handler, the `wait` command will stop the upload process if it is executed while the download is still going on. In particular, do not use constructions like the following, which will sit forever without uploading the file:

```
libURLftpUpload field "Upload Data",myURL  
wait until the URLStatus of myURL is "uploaded" -- DON'T DO THIS
```

The `URLStatus` function returns the status of the uploaded file. If you no longer need to monitor the file's status, use the `unload` command to remove it from the `URLStatus` function's listing.

Important! The `libURLftpUpload` command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `libURLftpUpload` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `libURLftpUpload` command can be used in any handler.

libURLftpUploadFile

command

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

ftp keyword, libURLDownloadToFile command, libURLftpUpload command, libURLSetFTPStopTime command, load command, put command, unload command, URLStatus function, How to cancel a file transfer in progress, Why can't I upload a file?

Summary

Uploads a file to an Internet server asynchronously via FTP.

Syntax

libURLftpUploadFile filePath,uploadURL[,callbackMessage]

Example Code

```
libURLftpUploadFile "/Disk/test.data","ftp://ftp.example.org/test"  
libURLftpUploadFile myFile,field "FTP URL","uploadDone"
```

Comments

Use the libURLftpUploadFile command to put a file on a server.

Parameters:

The filePath specifies the name and location of the file you want to upload. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

The uploadURL specifies the server and location to upload to, in the form of an ftp URL.

The callbackMessage is the name of a message to send after the file is uploaded.

Comments:

The libURLftpUploadFile command transfers the data directly from the file to the server. Unlike libURLftpUpload (or the put command used with an ftp URL), the data does not all need to be in memory at once, so this command is a better choice for large files.

The libURLftpUploadFile command transfers the file in binary mode.

The callbackMessage is sent to the object whose script contains the libURLftpUpload command, after the upload is complete, so you can handle the callbackMessage to perform any tasks you want to delay until the file has been uploaded. Two parameters are sent with the message: the URL and the URLStatus of the file.

Important! The libURLftpUploadFile command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the libURLftpUploadFile command is not yet available. This may affect attempts to use this command in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the libURLftpUploadFile command can be used in any handler.

libURLLastHTTPHeaders

function

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

http keyword, httpHeaders property command, libURLLastRHHeaders function, libURLSetCustomHTTPHeaders command, load command, URLStatus function, How to access the Internet from behind a firewall

Summary

Returns the value of the httpHeaders property used for the previous HTTP request.

Syntax

libURLLastHTTPHeaders()

Example Code

```
put libURLLastHTTPHeaders() into headersToAnalyze
if probReq is among the lines of libURLLastHTTPHeaders() then tryAgain
```

Comments

Use the libURLLastHTTPHeaders function to debug problems with HTTP transfers, or to report errors to advanced users.

Value:

The libURLLastHTTPHeaders function returns a string.

Comments:

You can set the httpHeaders property to send custom headers in addition to the default headers. Whenever Revolution contacts a web server to download a page (with the load command or by using a URL in an expression) or to post data (with the post command), the contents of the httpHeaders property is sent to the web server. The libURLLastHTTPHeaders function returns the last set of custom headers used.

Important! The libURLLastHTTPHeaders function is part of the Internet library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

libURLLastRHHeaders

function

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

http keyword, httpHeaders property command, libURLLastHTTPHeaders function, load command, URLStatus function, How to retrieve headers from an HTTP request

Summary

Returns the headers sent by the remote host in the most recent HTTP transaction.

Syntax

libURLLastRHHeaders()

Example Code

```
if libURLLastRHHeaders() is mostRecentSet then exit to top
```

Comments

Use the libURLLastRHHeaders function to respond to custom header information that the Internet library does not handle.

Value:

The libURLLastRHHeaders function returns a string.

Comments:

Whenever another system sends data via HTTP to your system, the remote system sends an HTTP header along with the data. This header generally consists of several lines and may include the request type, the HTTP protocol version being used, and other information.

Important! The libURLLastRHHeaders function is part of the Internet library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

libURLSetCustomHTTPHeaders

command

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

http keyword, httpHeaders property, libURLLastHTTPHeaders function, load command, URLStatus function, How to access the Internet from behind a firewall

Summary

Sets the headers to be sent with each request to an HTTP server.

Syntax

libURLSetCustomHTTPHeaders headersList

Example Code

```
libURLSetCustomHTTPHeaders "GET /catsdata/coldat.dat HTTP/1.1"  
libURLSetCustomHTTPHeaders field "Headers"
```

Comments

Use the libURLSetCustomHTTPHeaders command to implement an HTTP method other than GET, POST, DELETE, or PUT.

Parameters:

The headersList is a string, or an expression that evaluates to a string.

Comments:

Whenever Revolution contacts a web server (with the load command, the post command, or by using an http URL in an expression), the headersList is sent to the web server.

Usually, the headersList will include multiple lines. A typical headersList might look like the following example:

```
GET /catsdata/coldat.dat HTTP/1.1  
Connection: Close  
User-Agent: My Fancy Browser  
Host: www.cats-training.com  
Accept: image/gif, image/x-xbitmap, image/jpeg, image/png, */*
```

Accept-Encoding: gzip
Accept-Charset: iso-8859-1,*,utf-8

The `libURLSetCustomHTTPHeaders` setting takes effect for the next HTTP transaction. After the transaction, the headers are set back to the default. This means that if you want to do multiple HTTP transactions using the same set of custom headers, you must use the `libURLSetCustomHTTPHeaders` command before each transaction.

The `libURLSetCustomHTTPHeaders` setting replaces the Internet library's default headers. To add to the default headers instead of replacing them, use the `httpHeaders` property instead.

If you specify a set of headers using the `libURLSetCustomHTTPHeaders` command, the Internet library's default headers and the setting of the `httpHeaders` property are ignored, and the headers set by `libURLSetCustomHTTPHeaders` are used instead.

Important! The `libURLSetCustomHTTPHeaders` command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `libURLSetCustomHTTPHeaders` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `libURLSetCustomHTTPHeaders` command can be used in any handler.

For technical information about the standard headers recognized in the HTTP 1.1 protocol, see RFC 2616 at <<http://www.ietf.org/rfc/rfc2616.txt>>.

libURLSetFTPListCommand

command

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

ftp keyword, libURLftpCommand function, How to list the files in an FTP directory

Summary

Switches between sending LIST or NLST formats when listing the contents of an FTP directory.

Syntax

libURLSetFTPListCommand {"LIST" | "NLST"}

Example Code

```
libURLSetFTPListCommand "NLST"  
libURLSetFTPListCommand (the listCommand of this card)
```

Comments

Use the libURLSetFTPListCommand command to get a simple list of files in an FTP directory.

Comments:

A URL that ends with a slash (/) designates a directory (rather than a file). An ftp URL to a directory evaluates to a listing of the directory's contents. The format of a directory listing depends on which FTP command Revolution sends to the FTP server. You specify which command to use with the libURLSetFTPListCommand command.

If you specify LIST, directory listings are returned in the same format as the Unix "ls" command and include information such as permissions, owner, size, and last modification date as well as the name of each file or subdirectory. Use this format if you need the additional information about each item in the directory.

If you specify NLST, directory listings consists of a list of names of files and subdirectories, one per line, without the additional information provided in a LIST listing. Use this format if you need a simple list of files and don't want to parse the more complex LIST listing for the file names.

The list command is set to LIST when the application starts up.

Important! FTP servers are not uniform in their response to a request for a directory listing. Some servers may format directory listings differently from these descriptions. These are the most common formats, but they are not universal.

Important! The `libURLSetFTPListCommand` command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `libURLSetFTPListCommand` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `libURLSetFTPListCommand` command can be used in any handler.

libURLSetFTPMode

command

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

ftp keyword, libURLSetFTPStopTime command, libURLftpUpload command

Summary

Switches between active and passive mode for FTP transfers.

Syntax

```
libURLSetFTPMode {"active" | "passive"}
```

Example Code

```
libURLSetFTPMode "passive"  
libURLSetFTPMode (the hilitedButtonName of group "Mode")
```

Comments

Use the libURLSetFTPMode command to improve performance with FTP servers that require active mode.

Comments:

For most FTP servers, passive transfer mode will work without a problem. However, a few servers require transfers to be made in active mode. If you are having trouble with ftp commands with a particular server, try setting the mode to "active".

The FTP mode is set to passive when the application starts up. To use active mode exclusively, execute the following statement before making any ftp transfers:

```
libURLSetFTPMode "active"
```

Important! The libURLSetFTPMode command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `libURLSetFTPMode` command is not yet available. This may affect attempts to use this command in `startup`, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `libURLSetFTPMode` command can be used in any handler.

libURLSetFTPStopTime

command

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete URL command, ftp keyword, get command, libURLftpUpload command, put command

Summary

Sets the timeout value for FTP transfers.

Syntax

libURLSetFTPStopTime timeoutInSeconds

Example Code

```
libURLSetFTPStopTime 30 -- 30-second timeout  
libURLSetFTPStopTime field "Timeout Value"
```

Comments

Use the libURLSetFTPStopTime command to increase the efficiency of multiple FTP transfers.

Parameters:

The timeoutInSeconds is a non-negative integer, or an expression that evaluates to a non-negative integer.

Comments:

When Revolution opens a socket to a host in order to upload or download a file via ftp, or to delete a file, it leaves the connection open for a time after the transaction completes. If another file is requested from the same FTP server during that time, Revolution uses the same socket instead of opening another one. This saves time if you are transferring multiple files with the same server, since the socket does not need to be set up again for each file.

By default, the timeout value is 15 seconds.

Note: If the server closes the connection, Revolution does not attempt to keep it open. The timeout set by libURLSetFTPStopTime controls when Revolution closes the connection from its end, but if the server closes the connection, a longer timeout has no effect.

Important! The `libURLSetFTPStopTime` command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `libURLSetFTPStopTime` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `libURLSetFTPStopTime` command can be used in any handler.

libURLSetLogField

command

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

get command, ID property, long keyword, name keyword, post command, put command

Summary

Specifies a field for logging information about uploads and downloads.

Syntax

libURLSetLogField longFieldDescriptor

libURLSetLogField "none"

Example Code

```
libURLSetLogField "field 1"
```

```
libURLSetLogField (the
```

Comments

Use the libURLSetLogField command to debug file transfers.

Parameters:

The fieldDescriptor is any expression that evaluates to a field reference.

Important! The libURLSetLogField command does not accept direct field references. For example, the following statement causes an error message:

```
libURLSetLogField field "Log" -- CAN'T USE THIS FORM
```

Instead, use a form that evaluates to a field reference, like this:

```
libURLSetLogField the name of field "Log" -- use this form instead
```

```
libURLSetLogField ("field" && quote & "Log" & quote) -- or this
```

Comments:

During ftp and http transfers, Revolution logs the data sent from the server. If a log field has been set, this data is placed after the log field's contents.

To stop logging, use the following statement:

```
libURLSetLogField "none"
```

Important! The `libURLSetLogField` function is part of the Internet library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `libURLSetLogField` function is not yet available. This may affect attempts to use this function in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `libURLSetLogField` function can be used in any handler.

libURLSetStatusCallback

command

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

ftp keyword, http keyword, libURLDownloadToFile command, libURLftpUpload command, libURLftpUploadFile command, load command, URLStatus function

Summary

Sets up a callback message to be sent periodically during uploads and downloads.

Syntax

libURLSetStatusCallback [messageName,objectLongID]

Example Code

```
libURLSetStatusCallback "putPercentage",the long ID of me  
libURLSetStatusCallback myAction,the long ID of button "Upload"  
libURLSetStatusCallback -- turns off status callback messages
```

Comments

Use the libURLSetStatusCallback command if you want to do periodic tasks (such as updating a progress bar) during a data transfer.

Parameters:

The messageName is the name of a message to be sent whenever the URLStatus function is updated.

The objectLongID is the long ID of the object to send the message to.

Comments:

The Internet library periodically updates the URLStatus function during uploads and downloads made with the libURLDownloadToFile, libURLftpUpload, libURLftpUploadFile, put, and load commands, as well as downloads triggered by evaluating an ftp or http URL. Whenever the Internet library updates the URLStatus function, it also sends any callback message you've set up with the libURLSetStatusCallback command.

Callback messages are sent even during blocking uploads and downloads (such as a download started with the put command), so you can use this method to monitor progress for all file transfers.

The callback message is sent with two parameters: the URL and the URL's current status. For example, suppose you execute the following statement to set up a callback message:

```
libURLSetStatusCallback "myProgress",the long ID of button "Monitor"
```

Then suppose you begin an upload with the following command:

```
put field "Data" into URL "ftp://me:mypass@ftp.example.org/myfile"
```

Periodically while the text of the field is being uploaded to the file, a "myProgress" message is sent to button "Monitor". The first parameter sent with "myProgress" is the URL (in this case, the URL is "ftp://me:mypass@ftp.example.org/myfile"), and the second parameter is the current status of that URL. The "myProgress" handler might look like this:

```
on myProgress theURL,theStatus
  put theStatus into field theURL of stack "Status Monitor"
end myProgress
```

The URL status parameter is similar to the status returned by the URLStatus function, and is one of the following:

"queued": on hold until a previous request to the same site is completed
"contacted": the site has been contacted but no data has been sent or received yet
"requested": the URL has been requested
"loading,bytesReceived,bytesTotal": the URL data is being received
"uploading,bytesReceived,bytesTotal": the file is being uploaded to the URL
"downloaded": the application has finished downloading the URL
"uploaded": the application has finished uploading the file to the URL
"error": an error occurred and the URL was not transferred
"timeout": the application timed out when attempting to transfer the URL
empty: the URL was not loaded, or has been unloaded

Note: The third item (bytesTotal) in the "loading" or "uploading" status report is empty if it is not possible to determine the total file size. (For example, if an FTP server does not support the SIZE command, it's not possible to determine the file size when downloading a file from that server.)

If multiple file transfers are occurring at the same time, a separate callback message is sent for each URL. If you want your callback handler to treat URLs differently (for example, if you want to display separate progress bars for different URLs, or do one thing with http URLs and something else with ftp URLs), use a switch control structure in the handler.

You cannot have two callback messages set up at the same time for uploads and downloads. If you use the libURLSetStatusCallback command to set up a callback message, the Internet library forgets any callback message you previously set up with the command.

To turn off the callback message, use the following statement:

```
libURLSetStatusCallback -- no parameters: turns off the message
```

If you don't include the `messageName` and `objectLongID` parameters, the `libURLSetStatusCallback` command turns off the callback messages so that updating the `URLStatus` function does not send any messages.

Important! The `libURLSetStatusCallback` command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `libURLSetStatusCallback` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `libURLSetStatusCallback` command can be used in any handler.

libURLVersion

function

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revAppVersion function, systemVersion function, version function

Summary

Returns the version of the Internet library.

Syntax

libURLVersion()

Example Code

```
get libURLVersion()  
if libURLVersion is field "Latest" then go next card
```

Comments

Use the libURLVersion function to check whether the Internet library supports a given feature, or when reporting problems.

Value:

The libURLVersion function returns a string indicating the version of the Internet library.

Comments:

There is no special significance to the version number returned by libURLVersion.

Important! The libURLVersion function is part of the Internet library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the libURLVersion function is not yet available. This may affect attempts to use this function in startup, preOpenStack, openStack, or

preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the libURLVersion function can be used in any handler.

licensed

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

backScripts function, edit command, environment function, frontScripts function, insert script command, revLicenseType function, script property, scriptLimits function

Summary

Returns true if the development environment is running.

Syntax

the licensed
licensed()

Example Code

```
the licensed  
if not licensed() then hide button "Edit Script"
```

Comments

Use the licensed function to determine whether the user is using a standalone application or the development environment.

Value:

The licensed function returns true or false.

Comments:

If the licensed function is false, the current application is a standalone, which has certain limitations on scripting. These limitations are described by the scriptLimits function.

Note: If the user has the Evaluation Edition, the licensed function returns true.

line

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

character keyword, choose command, item keyword, lineDelimiter property, penColor property, token keyword, tool function, word keyword, About chunk expressions, Why is there a problem with line endings?, Object menu > New Control > Line Graphic

Summary

Designates a return-delimited string as part of a chunk expression. It also designates the paint tool used to draw line shapes and specifies, through the style property, that a graphic is a line.

Syntax

Example Code

```
get line 3 of the stacks
choose line tool
set the style of graphic "Divider" to "line"
```

Comments

Use the line keyword to paint a curved line with the penColor, to draw a line shape, or to refer to a specific line or lines in a container.

Comments:

When using the Line tool, the cursor is a crosshairs shape (over images) or an arrow (anywhere else). This tool is used to draw a straight line in the penColor.

If you try to paint with the Line tool on a card that has no images, an image the size of the card is created to hold the painting. If the current card already has one or more images, painting outside the image has no effect.

Setting the style of a graphic to "line" makes the graphic into a curved line. Line graphics, unlike painted lines, can be changed and reshaped: use the points property to change the line's angle and position.

Note: To avoid script errors, enclose the word "line" in double quotes when you use it to indicate a graphic's style. This prevents Transcript from getting confused about whether you mean the line style, or the contents of a line in a container.

By default, a line is delimited by return (ASCII 10). However, you can set the lineDelimiter property to any character in order to directly address chunks delimited by that character.

A line is a string of characters delimited by a single return character (or a single one of the lineDelimiter character). A single line can contain multiple characters, multiple words, and multiple items.

lineDelimiter

property

Synonyms

lineDel

Objects

local

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

formfeed constant, itemDelimiter property, line keyword, lineOffset function, lines keyword, return constant

Summary

Specifies the character used to separate lines in chunk expressions.

Syntax

set the lineDelimiter to character

Example Code

```
set the lineDelimiter to numToChar(13)
set the lineDelimiter to myRecordDelimiter
```

Comments

Use the lineDelimiter property to divide text into chunks based on a delimiting character.

Value:

The lineDelimiter is a character.

By default, the lineDelimiter property is set to return.

Comments:

The lineDelimiter is a single character. Chunk expressions use the lineDelimiter to determine where one line ends and the next begins.

Since the lineDelimiter is a local property, its value is reset to return when the current handler finishes executing. It retains its value only for the current handler, and setting it in one handler does not affect its value in other handlers it calls.

lineIncrement

property

Synonyms

lineInc

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

pageIncrement property, repeatRate property, scrollbarLineDec message, scrollbarLineInc message, thumbPosition property

Summary

Specifies how far a scrollbar scrolls when one of its arrows is clicked.

Syntax

set the lineIncrement of scrollbar to distance

Example Code

```
set the lineIncrement of scrollbar "Whole Page" to 100
```

Comments

Use the lineIncrement property to change the amount that is scrolled when the scrollbar arrows are clicked.

Value:

The lineIncrement of a scrollbar is a number between the scrollbar's startValue and endValue.

By default, the lineIncrement property of newly created scrollbars is set to 512.

Comments:

When the user clicks the arrows at the ends of a scrollbar, the scrollbar moves one line up or down (or to the left or right, for a horizontal scrollbar). Use the lineIncrement property to specify how far the scrollbar thumb moves.

The startValue and endValue properties set the scale used for the lineIncrement, so for example, if the lineIncrement is set to one-hundredth of the difference between the startValue and endValue, clicking an arrow moves the scrollbar one-hundredth of its length.

lineOffset

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

caseSensitive property, is among operator, itemOffset function, lineDelimiter property, offset function, wholeMatches property, wordOffset function

Summary

Returns the number of lines between the beginning of a value and an occurrence of a specified string.

Syntax

lineOffset(lineToFind,stringToSearch[,linesToSkip])

Example Code

```
lineOffset("cheese",foodsList)  
put subbedData into line (lineOffset(it,receivedData)) of receivedData
```

Comments

Use the lineOffset function to find which line a string occurs in.

Parameters:

The lineToFind is a string or an expression that evaluates to a string.

The stringToSearch is a string or an expression that evaluates to a string.

The linesToSkip is a non-negative integer. If you don't specify how many linesToSkip, the lineOffset function does not skip any lines.

Value:

The lineOffset function returns a non-negative integer.

Comments:

The value returned by the lineOffset function is the number of the line where lineToFind appears in stringToSearch. If the lineToFind is not in stringToSearch, the lineOffset function returns zero. If the lineToFind is more than one line, the lineOffset function always returns zero, even if the lineToFind appears in the stringToSearch.

If you specify how many `linesToSkip`, the `lineOffset` function skips the specified number of lines in the `stringToSearch`. The value returned is relative to this starting point instead of the beginning of the `stringToSearch`.

lines

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

characters keyword, is among operator, is not among operator, items keyword, line keyword, lineDelimiter property, lineOffset function, number function, sort command

Summary

Used with the sort command, number function, and is among and is not among properties to designate the return-delimited portions of a string.

Syntax

Example Code

```
put the number of lines of the result into linesToCheck
```

Comments

Use the lines keyword to sort or select individual lines.

Comments:

By default, a line is delimited by return (ASCII 10). However, you can set the lineDelimiter property to any character in order to directly address chunks delimited by that character.

A line is a string of characters delimited by a single return character (or a single one of the lineDelimiter character). A single line can contain multiple characters, multiple words, and multiple items.

lineSize

property

Synonyms

penHeight, penWidth

Objects

graphic, global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

arrowSize property, borderColor property, brushColor property, choose command, dashes property, filled property, tool property

Summary

Specifies the thickness of lines and the borders of shapes drawn with the paint tools.

Syntax

set the lineSize [of graphic] to widthInPixels

Example Code

```
set the lineSize to 3 -- 3-pixels lines will be used in paint tools
set the lineSize of graphic "Rectangular Border" to 9
```

Comments

Use the lineSize property to set the thickness of the line used for line, curve, and polygon shapes, and the outline used with the oval, rectangle, and roundRect shapes.

Value:

The lineSize of a graphic is an integer between zero and 32767.

By default, the lineSize of newly created graphics is set to 1.

Comments:

If the lineSize property is set to zero, the lines in the graphic disappear.

The global setting of the lineSize property controls the appearance of shapes drawn with the paint tools. Once a paint shape is drawn, its appearance cannot be changed by changing the global lineSize property.

Caution! Setting the lineSize of a graphic to a value greater than the height or width of the stack window may cause problems on Unix systems, and looks strange on any platform.

link

keyword

Synonyms

group

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickChunk function, group command, linkClicked message, linkColor property, linkHiliteColor property, linkText property, linkVisitedColor property, plain keyword, show groups command, textStyle property, How to create a hypertext link, Text menu > Link

Summary

Used with the textStyle property to make a run of text in a field behave like a single word when clicked.

Syntax

Example Code

```
set the textStyle of the foundChunk to link
```

Comments

Use the link keyword to create clickable text.

Comments:

Text with its textStyle set to "link" is treated specially by the clickText, clickChunk, mouseText, and mouseChunk functions: a style run of grouped text is treated as a single word. This makes grouped text handy to use for hypertext or "clickable text" features.

Note: Each continuous run of "link"-styled text is considered a single text group, even if you selected part of the text and set its textStyle to "link", and set the textStyle of the other part later. It is not possible for two separate text groups to exist if there are no characters between them.

To show which text in a field is grouped, it is underlined and displayed in the color specified by the linkColor property. Use the hide groups command to hide the underline.

Changes to Transcript:

The link keyword was introduced in version 1.1. In previous versions, the group synonym was used, and grouped text could be optionally displayed with a heavy gray underline.

linkClicked

message

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

click command, clickLoc function, linkText property, lockText property

Summary

Sent when the user clicks grouped text.

Syntax

linkClicked linkedTextOfChunk

Example Code

```
on linkClicked theText -- open the URL stored with the clicked text
  if theText is not empty then revGoURL theText
end linkClicked
```

Comments

Handle the linkClicked message to respond to the user clicking a text group.

Parameters:

The linkedTextOfChunk is the linkText property of the clicked text group.

Comments:

The linkClicked message is sent to the field that was clicked.

The linkClicked message is sent only when the Browse tool is being used and is sent only if the field is locked. The linkClicked message is not sent if the field's listBehavior property is true.

If the clicked text is grouped textóthat is, if its textStyle property contains "link"óthen the linkText property of the clicked group is sent as the linkedTextOfChunk parameter. If the text's linkText is empty, the clickText is sent as the linkedTextOfChunk parameter.

linkColor

property

Synonyms

Objects

global, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundColor property, colorNames function, colors property, foregroundColor property, link keyword, linkHiliteColor property, linkVisitedColor property, visited property, underlineLinks property, About colors and color references, Color Names Reference, Recipe for translating a color name to an RGB numeric triplet, Text menu > Link

Summary

Specifies the color of grouped text.

Syntax

set the linkColor to {colorName | RGBColor}

set the linkColor of stack to {empty | colorName | RGBColor}

Example Code

```
set the linkColor to "#99CC00"
```

```
set the linkColor of the mouseStack to the hiliteColor of me
```

Comments

Use the linkColor property to make grouped text look and behave like links in a web browser.

Value:

The linkColor is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the linkColor is set to "0,0,238" (blue). The linkColor of a newly created stack is set to empty by default.

Comments:

Visited text is text whose visited property is true. This property is set to true when the user has clicked a text group during the current session.

If the linkColor of a stack is empty, grouped text in that stack is shown with the global linkColor property if the text's visited property is false.

If the linkColor of a stack is not empty, unvisited grouped text in that stack is shown with the stack's linkColor property, regardless of the global setting. property, regardless of the global setting.

linkHiliteColor

property

Synonyms

Objects

global, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundColor property, colorNames function, colors property, foregroundColor property, link keyword, linkColor property, linkVisitedColor property, underlineLinks property, About colors and color references, Color Names Reference, Recipe for translating a color name to an RGB numeric triplet, Text menu > Link

Summary

Specifies the color of grouped text that is being clicked.

Syntax

set the linkHiliteColor to {colorName | RGBColor}

set the linkHiliteColor of stack to {empty | colorName | RGBColor}

Example Code

```
set the linkHiliteColor to "125,0,255"  
set the linkHiliteColor of this stack to myColor
```

Comments

Use the linkHiliteColor property to make grouped text look and behave like links in a web browser.

Value:

The linkHiliteColor is a color reference.

A colorName is any standard color name.

An RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the linkHiliteColor is set to "255,0,0" (bright red). The linkHiliteColor of a newly created stack is set to empty by default.

Comments:

The linkHiliteColor appears to show that grouped text is being clicked. When the mouse button is released, the text color changes to the linkVisitedColor.

If the linkHiliteColor of a stack is empty, grouped text in that stack is shown with the global linkHiliteColor property when being clicked.

If the linkHiliteColor of a stack is not empty, grouped text in that stack is shown with the stack's linkHiliteColor property, regardless of the global setting.

linkText

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

fixedLineHeight property, fontSizes function, printTextSize property, scriptTextSize property, textFont property, textHeight property, textShift property, textStyle property, Text menu > Size, Shortcut to remove font changes from text

Summary

Attaches data to a section of text in a field.

Syntax

set the linkText of chunk of field to textString

Example Code

```
set the linkText of line 2 to 3 of field "Contents" to "Leaves of Grass"
```

Comments

Use the linkText property to store the destination of a link, the text of an associated file, a definition, or other data along with a chunk of text in a field.

Value:

The linkText of a chunk is a string.

By default, the linkText property of any chunk is empty.

Comments:

The linkText property can be set for any chunk in a field.

However, it is most useful when used with grouped text. If the linkText of grouped text is set to a value, the value is passed with the linkClicked message when the user clicks the text. This lets you use the linkText to specify the destination of a link or other data to use when the text is clicked.

linkVisitedColor

property

Synonyms

Objects

global, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundColor property, colorNames function, colors property, foregroundColor property, link keyword, linkColor property, linkHiliteColor property, visited property, underlineLinks property, About colors and color references, Color Names Reference, Recipe for translating a color name to an RGB numeric triplet, Text menu > Link

Summary

Specifies the color of grouped text that has been clicked during the current session.

Syntax

set the linkVisitedColor to {colorName | RGBColor}

set the linkVisitedColor of stack to {empty | colorName | RGBColor}

Example Code

```
set the linkVisitedColor to "purple"  
set the linkVisitedColor of stack "Help" to empty
```

Comments

Use the linkVisitedColor property to make grouped text look and behave like links in a web browser.

Value:

The linkVisitedColor is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the linkVisitedColor is set to "81,24,128" (a dark purple). The linkVisitedColor of a newly created stack is set to empty by default.

Comments:

Visited text is text whose visited property is true. This property is set to true when the user has clicked a text group during the current session.

If the linkVisitedColor of a stack is empty, grouped text in that stack is shown with the global linkVisitedColor property if the text's visited property is true.

If the linkVisitedColor of a stack is not empty, already-visited grouped text in that stack is shown with the stack's linkVisitedColor property, regardless of the global setting.

listBehavior

property

Synonyms

autoSelect

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

autoHilite property, hilitedLine property, lockText property, nonContiguousHilites property, select command, selectedLine function, toggleHilites property, How to determine which line of a list field is selected, How to select an entry in a list field, Object menu > New Control > List Field

Summary

Specifies whether a locked field behaves as a clickable list.

Syntax

set the listBehavior of field to {true | false}

Example Code

```
set the listBehavior of field "Items List" to true
if the listBehavior of me then mouseUp
```

Comments

Use the listBehavior property to create a list box.

Value:

The listBehavior of a field is true or false.

By default, the listBehavior property of newly created fields is set to false.

Comments:

If a field's listBehavior property is set to true, and the user clicks a line, the entire line is highlighted. The Up and Down arrow keys move the selection up or down.

Normally, the mouseUp and mouseDown messages are sent to the field as usual. However, if the user clicks below the last line of text in the field, a mouseRelease message is sent instead of mouseUp.

You use the hilitedLine property to determine which line the user clicked. The field's hilitedLine is set to the new line before the mouseDown message is sent, so there is no way to determine the previously-selected line.

If the field's `autoHilite` property is set to `false`, a clicked line does not highlight, regardless of the `listBehavior` setting; the field does not behave like a clickable list in this case. However, you can set the `hilitedLine` of the field even if the field's `autoHilite` is `false`.

liveResizing

property

Synonyms

Objects

stack

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

rectangle property, resizable property, resizeStack message, revChangeWindowSize command, How to automatically adjust objects when a window is resized, How to respond to resizing a window

Summary

Causes a stack window to display its contents changing during resizing, instead of redrawing the contents after the window is resized.

Syntax

set the liveResizing of stack to {true | false}

Example Code

```
set the liveResizing of stack "Hello World" to false
set the liveResizing of me to true
```

Comments

Use the liveResizing property to create a smooth visual appearance while resizing.

Value:

The liveResizing property of a stack is true or false. By default, the liveResizing of a newly created stack is set to false.

Comments:

When the liveResizing property is true, the window contents (including the borders) are redrawn continuously as the user resizes, so at any time, the window is displayed as it will look if the user releases the mouse button at that moment.

If the liveResizing is false, the window does not change until the user releases the mouse and stops resizing.

If the liveResizing is true, resizeStack messages are sent continually while the window is being resized, allowing your stack to update its appearance during resizing. (If the user pauses during resizing, with the

mouse down in the resize box but the pointer not moving, no `resizeStack` message is sent until the mouse moves again.)

On Mac OS, Unix, and Windows systems, the `liveResizing` property has no effect.

ln function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

^ operator, exp function, ln1 function, log10 function, log2 function

Summary

Returns the natural logarithm of a number.

Syntax

the ln of number
`ln(number)`

Example Code

```
ln(1) -- returns zero  
ln(0.15) -- returns -1.89712  
ln(10) -- returns 2.302585
```

Comments

Use the ln function to obtain a base e logarithm.

Parameters:

The number is a positive number, or an expression that evaluates to a positive number.

Value:

The ln function returns a number.

Comments:

The natural logarithm of a number is the power to which e must be raised to obtain the number.

The transcendental number e appears in many mathematical formulas.

To find the logarithm of a number in any base, use the following function:

```
function logarithm theBase,theNumber  
    return ln(theNumber)/ln(theBase)
```

end logarithm

ln1

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

^ operator, exp1 function, ln function, log10 function, log2 function

Summary

Returns the natural logarithm of a number plus one.

Syntax

the ln1 of number

ln1(number)

Example Code

```
ln1(0) -- returns zero  
put ln1(myValue) into logValue
```

Comments

Use the ln1 function to obtain a base e logarithm, $\ln(\text{number} + 1)$.

Parameters:

The number is a number greater than -1, or an expression that evaluates to such a number.

Value:

The ln1 function returns a number.

Comments:

The natural logarithm of a number is the power to which e must be raised to obtain the number.

The transcendental number e appears in many mathematical formulas.

load

command

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cachedURLs function, get command, go command, httpHeaders property, httpProxy property, libURLftpUpload command, libURLSetCustomHTTPHeaders command, libURLSetStatusCallback command, revGoURL command, unload command, URL keyword, URLStatus function, How to access the Internet from behind a firewall, How to cancel a file transfer in progress, Why was a downloaded file corrupted?

Summary

Downloads the file specified by a URL to a cache where it can be used by another handler.

Syntax

load [URL] url [with message callbackMessage]

Example Code

```
load URL "http://www.example.com/index.html"  
load URL myURL with message "downloadComplete"
```

Comments

Use the load command to pre-fetch a file from the Internet in order to speed up access when using it in an expression with the URL keyword.

Parameters:

The url is any valid http or ftp URL.

The callbackMessage is the name of a message to send after the URL is loaded.

Comments:

To use a file that has been downloaded by the load command, refer to it using the URL keyword as usual. When you request the original URL, Revolution uses the cached file automatically.

The callbackMessage is sent to the object whose script contains the load command, after the URL is loaded, so you can handle the callbackMessage to perform any tasks you want to delay until the URL has been cached. Two parameters are sent with the message: the URL and the URLStatus of the file.

The load command is non-blocking, so it does not stop the current handler while the download is completed. The handler continues while the load command downloads the URL in the background. You can monitor the download by checking the URLStatus function periodically.

Caution! Avoid using the wait command in a handler after executing the load command. Since the load command is non-blocking, it may still be running when your handler reaches the wait command. And since the load command is part of the Internet library and is implemented in a handler, the wait command will stop the download process if it is executed while the download is still going on. In particular, do not use constructions like the following, which will sit forever without downloading the file:

```
load URL myURL
wait until the URLStatus of myURL is "cached" -- DON'T DO THIS
```

The file is downloaded into a local cache. It does not remain available after the application quits; the purpose of the cache is to speed up access to the specified URL, not to store it permanently. You can use a URL even if it is not in the cache, so use of the load command is optional.

Note: Cached files consume memory. To release this memory after you are finished with a URL, use the unload command to remove it from the cache.

Important! The load command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the load command is not yet available. This may affect attempts to use this command in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the load command can be used in any handler.

local

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, delete variable command, explicitVariables property, global command, localNames function, variableNames function, Recipe for 99 Bottles of Beer on the Wall (using send), Recipe for a "roll credits" effect

Summary

Declares one or more local variables and assigns initial values to them.

Syntax

local variablesList

Example Code

```
local currentStatus -- creates one local variable
local thisThing,thatThing,theOtherThing -- creates three variables
local A=1,B=2,C=3 -- creates variables with initial values
```

Comments

Use the local command to define a local variable for a handler, or to define a script local variable that is shared between all the handlers in a script.

Parameters:

The variablesList consists of one or more name=value pairs, separated by commas:

- The name is any string.
- The value is any literal string.

The value is optional; you can specify just the variable name. In this case, the specified local variables contain empty when created.

Comments:

The local declaration is optional. You can use a local variable without declaring it first in the handler. In this case, the local variable can be used only within that handler.

The local command can appear anywhere in a handler. However, variable declarations are usually placed at the beginning of a handler. This makes them easier to find.

You can also use the `local` command to create a script local variable. Script local variables can be used by any handler in that script, without needing to be declared in the handler itself, and their values are maintained from handler to handler. You create a script local variable by using the `local` command in a script, outside any handlers in the script. (The difference between a script local variable and a global variable is that a script local variable can only be used in the handlers of one script, while a global variable can be used in any handler or script with a global declaration for that variable.)

The value of a local variable is retained only while the handler is running. When the handler exits, the value of the local variable is lost.

The value of a script local variable is retained between handlers, but is lost when you quit the application, when you close the stack (unless its `destroyStack` property is `false`), or when the script is re-compiled.

localLoc

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickLoc function, foundLoc function, globalLoc function, mouseLoc function, screenMouseLoc property, selectedLoc function

Summary

Returns the equivalent, in local coordinates, of a point given in global coordinates.

Syntax

the localLoc of globalPoint

localLoc(globalPoint)

Example Code

```
localLoc(90,174)
```

Comments

Use the localLoc function to translate between screen and window coordinates.

Parameters:

The point is any expression that evaluates to a point—a vertical and horizontal distance from the top left of the screen, separated by a comma.

Value:

The localLoc function returns two integers separated by a comma.

Comments:

In window coordinates, the point 0,0 is at the top left of the stack window. In screen coordinates, the point 0,0 is at the top left of the screen.

The point returned by the localLoc function is relative to the top left of the current stack.

The first item of the returned value is the horizontal distance in pixels from the left edge of the current stack to the location given by point. The second item of the returned value is the vertical distance from the top edge of the current stack to the location given by point.

localNames

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

explicitVariables property, globalNames function, local command, variableNames function

Summary

Returns a list of all local variables.

Syntax

the localNames

localNames()

Example Code

the localNames

repeat with x = 1 to the number of items of line 2 of the localNames

Comments

Use the localNames function to determine which local variables are available to the current handler, or to make sure a local variable name has not already been used before declaring it.

Value:

The localNames function returns a value consisting of two lines:

1. Local variables created in the current handler
2. Local variables created in the current script but outside all handlers

Comments:

Local variables are created either implicitly, by putting a value into them, or explicitly with the local command. They can be deleted with the delete variable command.

location

property

Synonyms
loc

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
bottomLeft property, bottomRight property, globalLoc function, is within operator, localLoc function, lockLocation property, move command, moveControl message, rectangle property, revCacheGeometry command, revUpdateGeometry command, topLeft property, topRight property, How to automatically adjust objects when a window is resized, How to change the size and position of the video grabber window, How to respond to moving a window, Why aren't window positions saved in my application?, Recipe for centering a stack window on the screen

Summary
Specifies where an object is.

Syntax
set the location of object to point

Example Code
if the location of button 1 is within the rect of field 1 then selectIt
set the location of this stack to the mouseLoc

Comments
Use the location property to move an object without resizing it, or to find out where an object is.

Value:
The location of an object is any expression that evaluates to a pointótwo integers separated by a comma.

The first item of the location is the distance in pixels from the left edge of the screen (for stacks) or card (for other objects) to the center of the object. The second item is the distance in pixels from the top edge of the screen (for stacks) or card (for other objects) to the center of the object.

For cards, the location property is read-only and cannot be set.

Comments:

The location of a stack is in absolute (screen) coordinates. The first item of a card's location property is equal to the width of stack div 2; the second item is equal to the height of stack div 2. The location of a group or control is in relative (window) coordinates.

In window coordinates, the point 0,0 is at the top left of the stack window. In screen coordinates, the point 0,0 is at the top left of the screen.

Changing the location of an object moves it to the new position without resizing it. To change an object's size, set its height, width, or rectangle properties.

lock colorMap

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

lock cursor command, lock screen command, lockColorMap property, screenColors function, screenDepth function, unlock colorMap command

Summary

Sets the lockColorMap property to true, preventing the screen from being redrawn when the color table is changed.

Syntax

lock colorMap

Example Code

```
lock colorMap
if the platform is "MacOS" then lock colorMap
```

Comments

Use the lock colorMap command to prevent screen flashing when images and movies appear on a screen set to 256 or fewer colors.

Comments:

The lock colorMap command sets the lockColorMap property to true.

When all pending handlers are finished executing, the lockColorMap property is set back to false, undoing the lock colorMap command's action.

lock cursor

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

cursor property, lock moves command, lock screen command, lockCursor property, unlock cursor command

Summary

Sets the lockCursor property to true, preventing the cursor shape from changing.

Syntax

lock cursor

Example Code

```
lock cursor  
if the mouse is not down then lock cursor
```

Comments

Use the lock cursor command to make the cursor retain a specified appearance until you change it.

Comments:

The lock cursor command sets the lockCursor property to true.

Set the cursor property to change the cursor.

lock error dialogs

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

allowInterrupts property, errorDialog message, lock messages command, lock screen command, lockErrorDialogs property, unlock error dialogs command, Recipe for getting the dimensions of a picture file, Development menu > Suppress Errors

Summary

Sets the lockErrorDialogs property to true, preventing the error window from being displayed when an execution error occurs.

Syntax

lock error dialogs

Example Code

```
lock error dialogs
if the environment is among the lines of userEnv then lock error dialogs
```

Comments

Use the lock error dialogs command to handle execution errors in a custom handler, rather than allowing Revolution to display the standard error window.

Comments:

The lock error dialogs command sets the lockErrorDialogs property to true.

If an execution error occurs while the lockErrorDialogs property is set to true, an errorDialog message is sent to the object whose handler set the lockErrorDialogs to true.

When all pending handlers are finished executing, the lockErrorDialogs property is set back to false, undoing the lock error dialogs command's action.

lock menus

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

lock screen command, lockMenus property, unlock menus command

Summary

Sets the lockMenus property to true, preventing the menu bar from being updated when the controls for the menu bar are changed.

Syntax

lock menus

Example Code

```
lock menus  
if the lockScreen then lock menus
```

Comments

Use the lock menus command to hide changes in the menu bar from the user.

Comments:

The lock menus command sets the lockMenus property to true.

When all pending handlers are finished executing, the lockMenus property is set back to false, undoing the lock menus command's action. Any changes made to the menus become visible to the user.

lock messages

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

flushEvents function, lock error dialogs command, lock recent command, lock screen command, lockMessages property, unlock messages command, Development menu > Suppress Messages

Summary

Sets the lockMessages property to true, preventing setProp triggers, getProp calls, and certain messages from being sent.

Syntax

lock messages

Example Code

```
lock messages  
if the commandKey is down then lock messages
```

Comments

Use the lock messages command when a handler performs an action (such as opening a stack) and you want to speed up access by preventing unnecessary navigation messages (such as openStack) from being sent.

Comments:

The lock messages command sets the lockMessages property to true.

When all pending handlers are finished executing, the lockMessages property is set back to false.

lock moves

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

lock messages command, lock screen command, lockMoves property, move command, unlock moves command

Summary

Sets the lockMoves property to true, preventing motions caused with the move command from being immediately visible.

Syntax

lock moves

Example Code

```
lock moves
if the lockScreen is false then lock moves
```

Comments

Use the lock moves command to synchronize motions made with the move command.

Comments:

When you use the move command to move an object around the screen, the movement takes place immediately. If you want to move more than one object simultaneously, first use the lock moves command to prevent these motions from being seen. Then issue the necessary move commands to move the objects. Finally, use unlock moves to show the result of the move commands.

The lock moves command sets the lockMoves property to true.

When all pending handlers are finished executing, the lockMoves property is set back to false.

lock recent command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

go command, lock messages command, lock screen command, lockRecent property, recentCards property, recentNames property, unlock recent command

Summary

Sets the lockRecent property to true, preventing cards from being added to the recent cards list.

Syntax

lock recent

Example Code

```
lock recent  
if not the hilite of button "Advanced" then lock recent
```

Comments

Use the lock recent command to move among cards without disturbing the recent cards list.

Comments:

When either the user or a handler navigates to a card, that card is placed on the recent cards list, and the user can return to it with the go command or the "Go Recent" item in the View menu. If a handler visits a card that is not normally seen by the user, you can use the lock recent command to prevent the user from inadvertently going to that card later.

The lock recent command sets the lockRecent property to true. When all pending handlers are finished executing, the lockRecent property is set back to false.

lock screen

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

alwaysBuffer property, lock colorMap command, lock cursor command, lock menus command, lock moves command, lockScreen property, unlock screen command, Recipe for getting the dimensions of a picture file

Summary

Sets the lockScreen property to true, temporarily preventing the screen from being updated.

Syntax

lock screen

Example Code

```
lock screen  
if the shiftKey is not down then lock screen
```

Comments

Use the lock screen command to rearrange items on the screen or to change locations without the user seeing the transition.

Comments:

A handler may need to open a stack and then close it before the handler is completed, or to move or change the appearance of a number of objects on the screen. If the screen is locked before these changes occur, the user does not see the changes happen on screen. Locking the screen can prevent user confusion or unsightly screen flashing. It also increases the speed of the handler, since Revolution does not have to redraw all the intermediate states of the screen.

The lock screen command sets the lockScreen property to true. When all pending handlers are finished executing, the lockScreen property is set back to false and the user sees the current state of stack windows on the screen.

Note: When using script debug mode, the screen cannot be locked and the lock screen command has no effect.

lockColorMap

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

colormap property, lock colorMap command, lockScreen property, screenColors function, unlock colorMap command, Why does an image have the wrong colors?

Summary

Specifies whether changes to the color table cause the screen to be redrawn with the new color table.

Syntax

set the lockColormap to {true | false}

Example Code

```
set the lockColormap to true
```

Comments

Use the lockColorMap property to trade off between color accuracy and possible screen flashing when the bit depth of the screen is 8 bits (256 colors) or less.

Value:

The lockColormap is true or false.

By default, the lockColorMap property is set to true on Mac OS systems with 8-bit displays (the screenColors equal to 256), and to false otherwise.

Comments:

If the lockColorMap property is set to false, newly-displayed images, videoclips, and players can change the color table used by Revolution. This displays the colors in images and movies more accurately, but can cause screen flashing. (To minimize screen flashing when the lockColorMap is false, go to the card containing the image and then set the lockColorMap property to true.)

If the lockColorMap is true, the color table cannot be changed. In this case, if an image, video clip, or player contains a color not in the current color table, the closest existing color is substituted. This may cause banding or strange-looking displays in photographs and other images that contain a smooth gradation of colors.

This property has no effect on systems using 16-bit or 24-bit color.

lockCursor

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cursor property, defaultCursor property, lock cursor command, lockScreen property, unlock cursor command

Summary

Specifies whether the cursor shape changes depending on the current tool and what the mouse pointer is over.

Syntax

set the lockCursor to {true | false}

Example Code

```
set the lockCursor to true
```

Comments

Use the lockCursor property to set a custom cursor that persists after the handler exits.

Value:

The lockCursor is true or false.

By default, the lockCursor property is set to false.

Comments:

If the lockCursor property is set to true, the cursor shape can be set only by a handler, and does not change shape automatically as the mouse moves. (For example, the cursor normally turns into an arrow over a menu, an I-beam over an editable field, and so on.) If the lockCursor is false, the cursor automatically changes shape according to its location.

For example, to set a custom cursor while the mouse pointer is over a certain field, set the lockCursor to true and then set the cursor to the desired shape in a mouseEnter handler. In the field's mouseLeave handler, set the lockCursor to false to allow the cursor to automatically change shape.

Important! If the `lockCursor` is true, changes made by a handler to the cursor still take effect. This means that if a stack locks the cursor and a handler in another stack sets the cursor to another shape, the cursor changes to that shape and does not change back until the cursor is changed or the `lockCursor` is set to false. If you lock the cursor, make sure to unlock it as soon as the stack no longer needs it.

lockErrorDialogs

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

allowInterrupts property, errorDialog message, lock error dialogs command, lockMessages property, lockScreen property, try control structure, unlock error dialogs command, Development menu > Suppress Errors

Summary

Specifies whether execution errors cause an errorDialog message to be sent, or display the error window.

Syntax

set the lockErrorDialogs to {true | false}

Example Code

```
set the lockErrorDialogs to true
```

Comments

Use the lockErrorDialogs property to handle execution errors in a custom handler, rather than allowing Revolution to display the standard error window.

Value:

The lockErrorDialogs is true or false.

By default, the lockErrorDialogs property is set to false. It is reset to false when all pending handlers are finished executing.

Comments:

If an execution error occurs while the lockErrorDialogs property is set to false, the error window appears. This is the default behavior.

If an execution error occurs while the lockErrorDialogs property is set to true, an errorDialog message is sent to the object whose handler set the lockErrorDialogs to true.

If you set the `lockErrorDialogs` to `true`, you should provide an `errorDialog` handler to notify the user of the error and perform any necessary cleanup.

lockLocation

property

Synonyms

locked, lockLoc

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

height property, location property, lockCursor property, lockMessages property, lockMoves property, lockScreen property, revChangeWindowSize command, width property

Summary

Specifies whether the user can move a control.

Syntax

set the lockLocation of object to {true | false}

Example Code

```
set the lockLocation of button "Next" to true
set the lockLocation of group ID 4455 to false
```

Comments

Use the lockLocation property to protect a control from being moved by the user, or to change the result of setting the height or width property, or to prevent an image or player from changing its size to fit the contents.

Value:

The lockLocation of a control is true or false.

By default, the lockLocation property of a newly created control is set to false.

Comments:

If the lockLocation property of an object is false, the user can drag its handles with the Pointer tool to move or resize the object. If the object's lockLocation is true, the user cannot move it or resize it.

If the lockLocation of a player or image that references an external file is false, the image or player is resized to fit its contents whenever the card opens. If the lockLocation is true, the image retains whatever size you set, even if it's not the same size as the object's contents.

If the `lockLocation` of a group is `false`, the group is automatically resized when the objects in it are moved, resized, hidden, or shown. If the `lockLocation` is `true`, this automatic resizing does not occur, and objects that move outside the group's boundaries are not shown. If a group has a scrollbar, set its `lockLocation` to `true` to ensure that the group does not automatically resize to fit its contents.

If a control's `lockLocation` property is `false`, when you change its height, it shrinks or grows from the center: the control's top and bottom edges both shift, while its location property stays the same. If the control's `lockLocation` property is `true`, it shrinks or grows from the top left corner: the control's top edge stays in the same place, and the bottom edge moves.

The setting of the `lockLocation` property does not prevent a handler from moving or resizing an object.

lockMenus

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

lock menus command, lockScreen property, unlock menus command

Summary

Specifies whether the menu bar is updated when the buttons used for the menu bar are changed.

Syntax

set the lockMenus to {true | false}

Example Code

```
set the lockMenus to false
```

Comments

Use the lockMenus property to hide temporary menu bar changes from the user, or to "hold" changes until you've finished all of them and are ready to display the new menu bar.

Value:

The lockMenus is true or false.

By default, the lockMenus property is set to true. It is reset to true when all pending handlers are finished executing.

Comments:

If the lockMenus is set to true, changes in the menubar group do not appear in the menu bar itself. To make these changes appear on screen, the lockMenus must be set to false momentarily.

The setting of this property has no effect on Unix and Windows systems.

lockMessages

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

flushEvents function, lock messages command, lockRecent property, lockScreen property, unlock messages command, Development menu > Suppress Messages

Summary

Specifies whether setProp triggers, getProp calls, and certain built-in messages are sent automatically.

Syntax

set the lockMessages to {true | false}

Example Code

```
set the lockMessages to false
```

Comments

Use the lockMessages property to prevent unwanted handlers from being triggered, or to speed up operations when handlers that are normally run automatically are not needed.

Value:

The lockMessages is true or false

By default, the lockMessages property is set to false. It is reset to false when no handlers are executing.

Comments:

If the lockMessages property is set to true, the following are not sent:

- Navigation messages (such as openCard, closeStack, and so on)
- Object creation messages (such as newCard, newButton, and so on)
- getProp calls
- setProp triggers

It is useful to set this property if a handler temporarily visits a card and you don't want the normal messages to be triggered.

The `lockMessages` property is automatically set to `false` when a palette, modeless, or modal stack is opened, even if a handler is still running.

lockMoves

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

lock moves command, lockMessages property, lockScreen property, move command, unlock moves command

Summary

Specifies whether the motion of objects moved by the move command is seen immediately or delayed.

Syntax

set the lockMoves to {true | false}

Example Code

```
set the lockMoves to true
```

Comments

Use the lockMoves property to synchronize object movements.

Value:

The lockMoves is true or false.

By default, the lockMoves property is set to false. It is reset to false when no handlers are executing.

Comments:

If the lockMoves property is set to true, when you use the move command to move an object around the screen, the movement does not become visible until the lockMoves is set to false. You can set the lockMoves property to true, start several objects moving, then set the lockMoves to false to begin the motion, as in the following example:

```
on mouseUp
  set the lockMoves to true
  move button 1 to 300,200 in 2 seconds -- doesn't move yet
  move field 2 to 0,0 in 2 seconds -- doesn't move yet
  move graphic 3 to 0,400 in 2 seconds -- doesn't move yet
  set the lockMoves to false -- all three objects start moving
```

end mouseUp

If the lockMoves is false, the movement takes place immediately.

lockRecent

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

go command, lock recent command, unlock recent command

Summary

Specifies whether visited cards are added to the recent cards list used by go back, go forth, and go recent.

Syntax

set the lockRecent to {true | false}

Example Code

```
set the lockRecent to true
```

Comments

Use the lockRecent property to prevent the user from having easy access to a card used only by a handler.

Value:

The lockRecent is true or false.

By default, the lockRecent property is set to false. It is reset to false when no handlers are executing.

Comments:

If the lockRecent property is set to true, visits to a card are not recorded in the recent cards list.

The lockRecent property is automatically set to false when a palette, modeless, or modal stack is opened, even if a handler is still running.

lockScreen

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

lock screen command, lockCursor property, lockMenus property, lockMessages property, unlock screen command

Summary

Specifies whether changes to a stack appear on the screen.

Syntax

set the lockScreen to {true | false}

Example Code

```
set the lockScreen to (not the lockScreen)
```

Comments

Use the lockScreen property to control when changes to the appearance of objects appear to the user, or to speed up actions that change several objects.

Value:

The lockScreen is true or false.

By default, the lockScreen property is set to false. It is reset to false when no handlers are executing.

Comments:

If the lockScreen property is set to true, statements that affect the appearance of the stack windows—such as opening or closing a stack, going to another card, or changing properties of an object—are not shown on the screen until the lockScreen is set back to false.

If the lockScreen is false, all changes are displayed immediately.

Setting the lockScreen to true also speeds up handlers that affect the screen appearance. Since Revolution does not need to take time redrawing its windows when the lockScreen is true, a handler that visits several cards in a row or changes several objects runs faster if the lockScreen property is set to true while the changes are made.

The lockScreen property is automatically set to false when a palette, modeless, or modal stack is opened, even if a handler is still running.

Important! Open stacks cannot be brought to the front (using the go or topLevel command) while the lockScreen is true.

Transcript keeps count of how many times the screen has been locked. You must balance each unlock with a lock; if you lock the screen twice and then unlock it once, the screen remains locked. For example, the following pair of handlers draws everything while the display is still locked:

```
on mouseUp
  lock screen  -- first lock
  drawStuff  -- gets locked again and unlocked in drawStuff
  show image "Sprite"
  unlock screen -- now unlocked - 2 locks balanced by 2 unlocks
end mouseUp
```

```
on drawStuff
  lock screen  -- screen now locked twice
  show field "Notify"
  unlock screen -- not unlocked yet - locked twice, unlocked once
end drawStuff
```

Note: When using script debug mode, the screen cannot be locked and the setting of the lockScreen property has no effect.

lockText

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantModify property, disable command, lockScreen property, traversalOn property, How to allow copying text from a locked field, How to prevent entering certain characters in a field

Summary

Specifies whether the contents of a field can be selected and changed by the user.

Syntax

set the lockText of field to {true | false}

Example Code

```
set the lockText of the foundField to false
```

Comments

Use the lockText property to prevent users from changing the text in a field.

Value:

The lockText of a field is true or false.

By default, the lockText property of newly created fields is set to false.

Comments:

When a field's lockText property is false, the field can be edited: the user can select text, delete text, type, cut, copy, and paste. The cursor becomes an I-beam when the mouse pointer is over the field.

When the user clicks in the field, no mouseDown, mouseUp, mouseStillDown, mouseDoubleDown, or mouseDoubleUp messages are sent. (However, if the user Control-clicks or right-clicks, these messages are sent regardless of the field's lockText setting.)

When a field's lockText property is true, the user cannot edit the contents of the field. The cursor does not change when the mouse pointer is over the field, and clicking the field sends all normal mouse messages to it. (A handler can change the contents of a field with the put command and can select text using the select command, regardless of the lockText setting.)

If a field's `lockText` and `traversalOn` properties are both set to true, the user can select text, but not change it, and can scroll within the field using the keyboard. If the `lockText` is true and the `traversalOn` is false, the user can neither select nor edit the field's text. If the `lockText` is false and the `traversalOn` is true, the field can be edited.

log10

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

[^] operator, exp10 function, ln function, log2 function

Summary

Returns the base-10 logarithm of a number.

Syntax

the log10 of number
log10(number)

Example Code

```
log10(1000) -- returns 3, because 10^3 = 1000  
log10(0.1) -- returns -1  
log10(163)
```

Comments

Use the log10 function to obtain a logarithm.

Parameters:

The number is a positive number, or an expression that evaluates to a positive number.

Value:

The log10 function returns a number.

Comments:

The logarithm of a number is the power to which 10 must be raised to obtain the number:
 $10^{\log_{10}(\text{number})}$ is equal to number.

log2

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

\wedge operator, exp2 function, ln function, log10 function

Summary

Returns the base-2 logarithm of a number.

Syntax

the log2 of number
`log2(number)`

Example Code

```
log2(8) -- returns 3  
log2(0.125) -- returns -3
```

Comments

Use the log2 function to obtain a base-2 logarithm.

Parameters:

The number is a positive number, or an expression that evaluates to a positive number.

Value:

The log2 function returns a number.

Comments:

The base-2 logarithm of a number is the power to which 2 must be raised to obtain the number:
 $2^{\log_2(\text{number})}$ is equal to number.

long
keyword

Synonyms
detailed

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

abbreviated keyword, convert command, date function, english keyword, files function, folders function, ID property, name property, short keyword, time function

Summary

Specifies a format for the date, time, files, and folders functions, the convert command, and the name, ID, and owner properties.

Syntax

Example Code

```
put the long ID of this card after cardsList
if field "Last Modified" is the long date then put true into itsToday
put getSizes(the detailed files) into field "Sizes"
```

Comments

Use the long keyword to obtain a date, time, name, ID, or list of files or folders with the greatest amount of available detail.

Comments:

In general, a long form is longer than either the short form or the abbreviated form.

If the useSystemDate property is set to false, a long date looks like this:

Thursday, June 22, 2000

If the useSystemDate is true, the long date is formatted according to the system settings.

If the useSystemDate property is set to false, a long time looks like this:

11:22:47 AM

If the useSystemDate is true, the long time is formatted according to the system settings.

A long object name consists of the object type, followed by the name in quotes, for the object and its owner (and the owner of that object, and so on). For example, a long card name looks like this:

card "My Card" of stack "/Drive/Folder/Stack.rev"

A long object ID looks like this:

field ID 2238 of stack "/Drive/Folder/Stack.rev"

Note: The long keyword is implemented internally as a property and function, and appears in the `propertyNames` and the `functionNames`. However, it cannot be used as a property in an expression, nor with the `set` command.

Changes to Transcript:

The forms the detailed files and the detailed folders were introduced in version 1.1. In previous versions, the files and folders functions supported only a simple list of file or folder names.

The detailed synonym was introduced in version 1.1.

The form the long owner was introduced in version 2.0. In previous versions, the form of the owner property could not be specified and reported only the abbreviated owner.

longFilePath

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

filename property, shortFilePath function, About filename specifications and file paths

Summary

Returns the long-format equivalent of an 8.3-format file path.

Syntax

the longFilePath of filePath

longFilePath(filePath)

Example Code

```
the longFilePath of it  
if storedPath is not the longFilePath of storedFile then checkVersion
```

Comments

Use the longFilePath function to get the long equivalent of a short file path passed from the command line.

Parameters:

The filePath is the location and name of the file or folder whose long equivalent you want. If you specify a name but not a location, Revolution assumes the file or folder is in the defaultFolder.

Value:

The longFilePath function returns a file path.

Comments:

The longFilePath function transforms the filePath without looking for the file, so it returns a value even if the filePath does not exist.

On Mac OS, OS X, or Unix systems, the longFilePath function simply returns the filePath without changing it.

longWindowTitles

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

decorations property, label property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the longWindowTitles to {true | false}

Example Code

Comments

In HyperCard, the longWindowTitles property determines whether a stack window's title bar shows the full path to a stack, or only the stack's name. In Revolution, the stack's label property is shown as the window title; if the label is empty, the short version of the stack's name property is shown.

A handler can set the longWindowTitles without causing a script error, but the actual window title is not changed. The default value is false.

lookAndFeel

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

environment function, hidePalettes property, machine function, platform function, pointerFocus property, raisePalettes property, How to change behavior depending on the platform, How to preview how a stack will look on other platforms, Why doesn't my application look like other Mac OS applications?, View menu > Look and Feel

Summary

Specifies which platform the user-interface controls resemble.

Syntax

set the lookAndFeel to {"Appearance Manager"|Macintosh|Motif|"Windows 95"}

Example Code

```
set the lookAndFeel to "Motif"
set the lookAndFeel to the selectedLine of field "Appearance"
```

Comments

Use the lookAndFeel property to preview the appearance of stacks on a platform other than the one you're developing on.

Value:

The lookAndFeel is one of the following:

Appearance Manager: standard Mac OS and OS X look and feel

Macintosh: emulated Mac OS Platinum appearance, used regardless of current theme

Motif: standard Motif look and feel for Unix systems

Windows 95: standard Windows look and feel

By default, the lookAndFeel is set to the platform the stack is being used on. On Mac OS and OS X systems, the lookAndFeel is set to "Appearance Manager" by default.

Comments:

The lookAndFeel property determines the appearance and behavior of scrollbars, object borders, checkboxes and radio buttons, and button menus. It also changes the appearance of the active (focused) control.

However, changing this property does not provide an exact representation of the appearance and behavior of the stack on the target platform. For example, cursors do not change, and neither do the placement of the menu bar or the way window dragging and resizing works. Only the appearance of controls is affected.

The "Appearance Manager" option can be used only on Mac OS and OS X systems. If you set the lookAndFeel to "Appearance Manager" on a Unix or Windows system, it is reset to "Macintosh".

On Mac OS systems, the native Appearance Manager drawing routines are much slower than the emulated Platinum routines. Setting the lookAndFeel to "Macintosh" rather than "Appearance Manager" will speed up drawing of controls.

Note: The phrases "Appearance Manager" and "Windows 95" must be enclosed in quotes because they consist of more than one word.

Changes to Transcript:

The "Appearance Manager" option was introduced in version 1.1. In previous versions, Revolution applications always used the Platinum appearance on Mac OS systems, regardless of which theme was selected on the user's system.

looping

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

duration property, endTime property, play command, playRate property, repeatCount property, start command, startTime property, stop command

Summary

Specifies whether a movie or sound stops at the end, or restarts from the beginning.

Syntax

set the looping of player to {true | false}

Example Code

```
set the looping of last player to true
```

Comments

Use the looping property to set a movie or sound to continuous play.

Value:

The looping of a player is true or false.

By default, the looping property of newly created players is set to false.

Comments:

If a player's looping property is set to true, when the movie or sound is played, it repeats over and over until stopped by the user or a handler. If the looping is false, the sound or movie plays through only once, then stops.

If the player's playSelection property is true, only the selected portion of the movie or sound is played.

lowResolutionTimers

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

move command, queryRegistry function, wait command

Summary

Specifies which operating-system mechanism is used to compute times on Windows systems.

Syntax

set the lowResolutionTimers to {true | false}

Example Code

```
set the lowResolutionTimers to true
```

Comments

Use the lowResolutionTimers property to improve performance on Windows systems.

Value:

The lowResolutionTimers is true or false.

By default, the lowResolutionTimers property is set to false.

Comments:

If the lowResolutionTimers property is set to true, timing intervals are measured with the operating system's WM_TIMER messages. If the lowResolutionTimers is false, timing intervals are measured with the high-precision multimedia timers.

The high-precision timers are sometimes unreliable, causing delays in the action of the move and wait commands. You can fix these symptoms by setting the lowResolutionTimers to true. However, using the WM_TIMER messages may cause jerkiness in the action of the move command.

The setting of this property has no effect on Mac OS and Unix systems.

LPT1:
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems
Not supported on Mac OS X systems
Supported on Windows systems
Not supported on UNIX systems

See Also:

close file command, COM1: keyword, COM2: keyword, COM3: keyword, COM4: keyword, COM5: keyword, COM6: keyword, COM7: keyword, COM8: keyword, COM9: keyword, modem: keyword, open file command, printer: keyword, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands to specify the line printer (parallel) port on Windows systems.

Syntax

Example Code

```
write myData to file "LPT1:"
```

Comments

Use the LPT1: keyword to communicate through the parallel port.

Comments:

To read data from the parallel port, use the read from file command, specifying the keyword LPT1: as the file to read from.

To write data to the parallel port, use the write to file command.

LZWKey

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

export command

Summary

Reserved for internal use.

Syntax

Example Code

Comments

machine function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

environment function, platform function, processor function, Supported Platforms Reference

Summary

Returns the type of hardware the application is running on.

Syntax

the machine
machine()

Example Code

```
the machine  
if the machine contains "Powerbook" then checkBattery
```

Comments

Use the machine function to change what a handler does based on the type of system it's running on.

Value:

The machine function returns a string.

Comments:

On Mac OS systems, the machine function uses the "Gestalt()" system call to determine the machine type.

On Unix systems, the machine function uses the "uname()" system function.

On Windows systems, the machine function always returns "x86".

macToISO

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

charSet property, charToNum function, ISOtoMac function, numToChar function, platform function

Summary

Returns the equivalent of a Mac OS character-set string, translated into the ISO 8859-1 character set.

Syntax

the macToISO of macString

macToISO(macString)

Example Code

```
macToISO("ISO 8859-1") -- returns "ISO 8859-1" unchanged  
macToISO("Áedilla") -- returns "¡edilla"
```

Comments

Use the macToISO function to translate data that was created on a Mac OS system to the ISO 8859-1 character set used on Unix and Windows systems.

Parameters:

The macString is any string.

Value:

The macToISO function returns the macString, with characters whose ASCII value is greater than 127 converted to their equivalent in the ISO 8859-1 character set. Characters whose ASCII value is less than 128 are left unchanged.

Comments:

Revolution automatically translates text in fields and scripts, as well as the names of custom properties, into the appropriate character set when you move a stack from one platform to another. It is therefore not necessary to translate them. However, the contents of custom properties, since they may contain binary data, are not translated automatically and must be translated if they contain characters whose ASCII value is 128 or greater.

Characters whose ASCII value is less than 128 are the same in the Mac OS character set and ISO 8859-1, so they are not changed by the macToISO function. These characters include uppercase and lowercase letters, numbers, and most punctuation.

magnifier

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

magnify property

Summary

Reserved for internal use.

Syntax

Example Code

Comments

magnify

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

choose command, hotSpot property, rectangle property, tool property, Shortcut to magnify an image

Summary

Shows or hides a window showing a magnified view of an image.

Syntax

set the magnify of image to {true | false}

Example Code

```
set the magnify of image "Dots" to true
```

Comments

Use the magnify property to edit an image pixel-by-pixel.

Value:

The magnify of an image is true or false.

Comments:

When you set the magnify of an image to true, a palette appears displaying a close-up view of the center of the image. At the same time, a box appears in the stack window showing you what portion of the image is magnified. Drag this box by its edges to magnify different parts of the image. In the Magnify palette, you use the paint tools to change the image.

Setting the magnify of the image to false hides the Magnify palette.

You can also show the Magnify palette by choosing the Pencil tool and Command-clicking the image (on Mac OS systems) or Control-clicking it (on Unix and Windows systems). This action selects the image, so if the image is part of a group, either the editGroupedControls property must be true, or you must be in group-editing mode.

mainStack

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

mainStacks function, openStacks function, revert command, save command, substacks property, About main stacks, substacks, and the organization of a stack file, File menu > New Substack of (main stack name), File menu > Move Substack to File...

Summary

Specifies which main stack a substack belongs to.

Syntax

set the mainStack of stack to mainStack

Example Code

```
set the mainStack of this stack to "Central"  
go the mainStack of stack "/Disk/Folder/file.rev"  
set the mainStack of stack "Hello" to "Goodbye"
```

Comments

Use the mainStack property to organize stacks in files.

Value:

The mainStack of a stack is a string, which is the short name of a stack.

By default, the mainStack property of newly created stacks is set to the new stack itself.

Comments:

Each Revolution file contains either a single main stack, or a main stack and one or more substacks. You can use this capability to bundle several related stacks into a single file for easy distribution, or to organize the stacks into categories, or to allow several stacks to inherit properties from the same main stack.

The mainStack of a stack file is the main stack in that file. (Each stack file can have only one main stack.)

The mainStack of a main stack is itself.

The mainStack of a substack is the main stack in that substack's file.

Changing a stack's mainStack property moves it into the same file as the specified main stack. The stack becomes a substack of the specified mainStack.

Changing a substack's mainStack property to itself makes the substack into an independent main stack (and removes it from the original main stack's substacks property). The next time you save the stack, Revolution will prompt you for a file name and location for the stack.

You can set the mainStack property of a stack to the name of any open stack.

Note: The mainStack property is simply the short name of the main stack, not a full stack reference.

mainStackChanged

message

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

mainStack property

Summary

Sent to a stack when its mainStack is changed.

Syntax

mainStackChanged

Example Code

```
on mainStackChanged -- in a main stack's script
  save me -- save the stack file when a substack is added
end mainStackChanged
```

Comments

Handle the mainStackChanged message if you want to respond when a substack is moved to another stack file.

Comments:

Each stack file consists of a main stack, and may also include one or more substacks. To move a stack to another file, you set its mainStack property to the name of the main stack of the destination file. The moved stack becomes a substack of the main stack you specified.

The mainStackChanged message is sent right after the stack is moved to the new stack file. This means that (if the substack does not trap the message) the mainStackChanged message is received by the new main stack, and can be handled in the main stack's script.

The mainStackChanged message is used by the Revolution development environment, so if you handle this message, be sure to pass it at the end of the handler.

mainStacks

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

defaultStack property, mainStack property, name property, openStacks function, revLoadedStacks function, stacks function, substacks property, topStack function

Summary

Returns a list of the main stacks that are loaded into memory.

Syntax

the mainStacks

mainStacks()

Example Code

```
the mainStacks
```

```
repeat while myStack is among the lines of the mainStacks
```

Comments

Use the mainStacks function to find out which main stacks are currently available.

Value:

The mainStacks function returns a list with the short name property of each main stack, one stack per line.

Comments:

Stacks that have not yet been saved are included in the list.

The list includes stacks that are part of the Revolution development environment (such as the message box and menu bar).

margins

property

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomMargin property, boundingRect property, formattedHeight property, formattedWidth property, leftMargin property, rectangle property, rightMargin property, topMargin property, wideMargins property

Summary

Specifies how much empty space is left on each side between an object's edges and its contents.

Syntax

set the margins of object to {pixels | left,top,right,bottom}

Example Code

```
set the margins of group ID 8 to 20 -- sets all four margins
set the margins of button myButton to 2,10,2,10
set the margins of field "Info" to three
```

Comments

Use the margins property to control how close text within a button or field can come to the object's edges, how close objects in a group can come to the group's edges, and how the label of a graphic is displayed.

Value:

The margins of an object consists of either a non-negative integer, or four non-negative integers separated by commas.

By default, the margins property of a field is set to 8. If the field's wideMargins property is true, the field's margins is set to 20.

The default margins setting for an object other than a field is 4.

Comments:

If a single integer is specified, all four margins of the object are set to that number of pixels. If four integers are provided, the object's top, left, bottom, and right margins are set to each one respectively:

- item 1 of the margins property is equal to the leftMargin
- item 2 of the margins is equal to the topMargin
- item 3 of the margins is equal to the rightMargin
- item 4 is equal to the bottomMargin

If the lookAndFeel is set to "Motif", controls require two pixels of margin space for the border that shows when the control is active (focused). To avoid interfering with this border on Unix systems, set the margins property of groups to at least 2 if the group contains controls whose traversalOn property is set to true.

Important! The margins of an object include the 2-pixel space required for the focus border, even if the lookAndFeel is not "Motif". This means that 2 is the smallest usable margin, rather than zero. For example, if the margins of a field is set to zero, a few pixels at the edge of the field's text may be cut off.

The margins setting of an image, player, or scrollbar has no effect.

mark

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

find command, go command, mark property, print command, sort command, unmark command

Summary

Sets the mark property of one or more cards to true.

Syntax

mark {card |all cards | cards where condition}
mark cards by finding findString [in field]

Example Code

```
mark card "Good & Plenty"  
mark cards where field "Intro" is empty  
mark cards where the short name of this card is not "Template"  
mark cards by finding "?" in field "Nationality"
```

Comments

Use the mark command to specify certain cards for printing or for special treatment by the go and sort commands.

Parameters:

The card is any card reference.

The condition is an expression that evaluates to true or false for each card in the current stack. Any object references within the condition are treated as pertaining to each card being evaluated, so for example, a reference to a field is evaluated for each card.

The findString is an expression that evaluates to a string.

The field is any field reference.

Comments:

You can mark cards either one at a time, by specifying each card to be marked, or in groups, by using the where condition or by finding forms of the mark command. Marking additional cards does not change the mark property of cards that have already been marked.

You use the mark and unmark commands in succession to further narrow down the set of cards. For example, this sequence of commands selects only the cards where the user has asked for help and the request has not yet been fulfilled:

```
mark cards by finding "Help" in field "History"  
unmark cards where the hilite of button "Already Helped" is true
```

When using the by finding form, the search operates the same way the normal form of the find command does. The mark command searches for cards that contain each word in the findString. The words in the findString must be found at the beginning of a word on the card, but the words do not need to be found together.

mark

property

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

find command, go command, mark command, sort command, unmark command

Summary

Specifies whether a card has been marked for later access by the sort, go, or print commands.

Syntax

set the mark of card to {true | false}

Example Code

```
set the mark of card 1 to false
```

Comments

Use the mark property to select a subset of cards for sorting, printing, or further searching.

Value:

The mark of a card is true or false.

By default, the mark property of newly created cards is set to false.

Comments:

The mark property of a card can be used to narrow down the action of several commands. For example, you can use the go command in forms such as go to next marked card. The sort command can be restricted to sorting only marked cards.

You can also use the mark and unmark commands to change a card's mark property.

This property is the equivalent of the "marked" property in HyperCard.

markChar

property

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

commandChar property, menuItem keyword

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the markChar of menuItem to character

Example Code

Comments

In HyperCard, the markChar property places a character (such as a checkmark) next to a menu item.

A handler can set the markChar to any value without causing a script error, but the actual appearance of the menu item is not changed.

marked

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

mark command, mark property, unmark command, About object types and object references

Summary

Used in card references and by the sort and print commands.

Syntax

Example Code

```
go to marked card 3
get the short name of next marked card
sort marked cards descending by field "Name"
print marked cards
```

Comments

Use the marked keyword to refer to cards whose mark property is true.

Comments:

You use the mark command to set the marked property of a subset of cards. Once marked, the cards can be treated as a set by the sort or print command. The form sort marked cards sorts only the cards whose mark property is true. The form print marked cards prints only the cards whose mark property is true.

In card references, the marked keyword indicates a card selected from the subset of marked cards. For example, the following statement goes to the first card in the current stack whose mark property is true:

```
go first marked card
```

markerDrawn

property

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

borderColor property, hiliteColor property, markerFilled property, markerLineSize property, markerPoints property, points property, visible property

Summary

Specifies whether markers are drawn at each vertex of a curve or irregular polygon graphic.

Syntax

set the markerDrawn of graphic to {true | false}

Example Code

```
set the markerDrawn of me to true
```

Comments

Use the markerDrawn property to draw a small shape at each vertex of a graphic.

Value:

The markerDrawn of a graphic is true or false.

By default, the markerDrawn property of newly created graphics is set to false.

Comments:

Irregular polygon graphics can be drawn with a marker at each vertex, whose shape is defined by the markerPoints property. The markerDrawn property specifies whether these markers are visible or not.

The borderPattern and borderColor properties determine the appearance of the markers' interior. The hilitePattern and hiliteColor determine the appearance of the borders.

If the style property of the graphic is not polygon or curve, the setting of its markerPattern property has no effect.

If the markerPoints property is empty, the setting of the markerDrawn property has no effect.

markerFilled

property

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

borderColor property, borderPattern property, filled property, markerDrawn property, markerLineSize property, markerPoints property

Summary

Specifies whether the markers drawn at each vertex of a polygon graphic are filled or hollow.

Syntax

set the markerFilled of graphic to {true | false}

Example Code

```
set the markerFilled of graphic 10 to false
```

Comments

Use the markerFilled property to change the appearance of the small shapes at each vertex of a graphic.

Value:

The markerFilled of a graphic is true or false.

By default, the markerFilled property of newly created graphics is set to false.

Comments:

Irregular polygon graphics can be drawn with a marker at each vertex, whose shape is defined by the markerPoints property. If the markerFilled property is true, the marker shapes are filled with the borderPattern or borderColor. If it is false, the marker shapes are hollow.

If the style property of the graphic is not polygon or curve, the setting of its markerFilled property has no effect.

If the markerPoints property is empty or the markerDrawn property is false, the markerFilled property has no effect.

markerLineSize

property

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hiliteColor property, hilitePattern property, lineSize property, markerDrawn property, markerPoints property

Summary

The markerLineSize specifies the thickness of the border of markers that are drawn at each vertex of an irregular polygon graphic.

Syntax

set the markerLineSize of graphic to pixels

Example Code

```
set the markerLineSize of last graphic to 0 -- no border
```

Comments

Use the markerLineSize property to change the appearance of the small shapes at each vertex of a graphic.

Value:

The markerLineSize of a graphic is a non-negative integer.

By default, the markerLineSize property of newly created graphics is set to 1.

Comments:

Irregular polygon graphics can be drawn with a marker at each vertex. The markerLineSize property specifies the width of the line that outlines each marker. The marker's shape is determined by the markerPoints property, and its outline color is determined by the hiliteColor property.

If the style property of the graphic is not polygon or curve, the setting of its markerPattern property has no effect.

If the markerPoints property is empty or the markerDrawn property is false, the markerLineSize property has no effect.

markerPoints

property

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hiliteColor property, markerDrawn property, markerFilled property, points property

Summary

Specifies the shape of markers at each vertex of a curve or polygon graphic.

Syntax

set the markerPoints of graphic to listOfVertexes

Example Code

```
set the markerPoints of the target to savedMarkerShape
set the markerPoints of me to 0,0 & return & 0,3 & return & 3,3
    & return & 3,0 & return & 0,0 -- defines a small square marker
```

Comments

Use the markerPoints property to create a small shape to mark the vertexes of a graphic.

Value:

The markerPoints of a graphic is a list of points (two integers separated by a comma), one per line.

By default, the markerPoints property of newly created graphics is set to empty.

Comments:

Curve and irregular polygon graphics can be drawn with a marker at each vertex. The marker itself is in the shape of a polygon, and the markerPoints property specifies this shape by specifying each vertex of the marker. Each point consists of the horizontal distance in pixels from the left edge of the marker to the marker vertex, a comma, and the vertical distance in pixels from the top edge of the marker to the marker vertex.

The first line in the list is the location of the marker shape's starting point. A blank line in the markerPoints indicates that the previous and next vertexes are not connected by a line—that is, the marker is broken into two (or more) pieces.

If the style property of the graphic is not polygon or curve, the setting of its markerPoints property has no effect.

If the markerDrawn property is false, the markerPoints property has no effect.

maskData

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

alphaData property, imageData property, maskPixmapID property, opaque property, windowShape property

Summary

Specifies which pixels of an image are displayed.

Syntax

set the maskData of image to binaryData

Example Code

```
put the maskData of image "Download" into dataToAnalyze
```

Comments

Use the maskData property to change the appearance of an image without changing its image data, or to examine the mask of the image.

Value:

The maskData of an image consists of a sequence of binary values.

Comments:

Each pixel is represented by 8 bits (1 byte) of mask data, with pixels numbered from the top left corner of the image, left to right, then top to bottom.

Since each pixel is represented by 8 bits (1 byte or 1 character), you can obtain the numeric value for a given pixel using the charToNum function. For example, the numeric value of the maskData for the tenth pixel is given by the expression charToNum(char 10 of the mask of image).

A value of zero means the pixel is fully transparent; any other value means the pixel is fully opaque. Unlike the alphaData property, the maskData stores only complete transparency or complete opacity, and does not support partial transparency.

maskPixmapID

property

Synonyms

Objects

image

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

export command, externals property, imagePixmapID property, import command, pixmapID property, windowID property, windowShape property

Summary

Specifies the window ID of the X11 pixmap used to mask an image.

Syntax

set the maskPixmapID of image to {empty | windowID}

Example Code

```
set the maskPixmapID of image ID 4500 to empty
```

Comments

Use the maskPixmapID property to pass to an external that needs to manipulate the image mask.

Value:

The maskPixmapID of a stack is an integer.

By default, the maskPixmapID of newly created images is empty.

Comments:

The maskPixmapID is provided by the operating system and indicates the pixmap used to mask an image. The mask is a 1-bit image map. Pixels in the image that correspond to a value of zero in the mask are transparent to mouse clicks, and pixels corresponding to a value of 1 can receive mouse clicks. If the maskPixmapID is empty, all the pixels in the image can receive mouse clicks (as though the mask consisted of all 1s).

This property is not supported on Mac OS and Windows systems.

matchChunk

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

filter command, format function, matchText function, offset function, wholeMatches property, Regular Expressions Syntax Reference

Summary

Returns true if a regular expression is found in the specified string, false otherwise.

Syntax

matchChunk(string,regularExpression[,positionVarsList])

Example Code

```
matchChunk("Hello","Hel") -- returns true
matchChunk("Hello","o$") -- also returns true
matchChunk(the short name of this stack,"^rev") -- starts with "rev"
matchChunk(field "Title",myExpr,startMatch,endMatch)
```

Comments

Use the matchChunk function to check whether a string contains a specified pattern.

Parameters:

The string is any expression that evaluates to a string.

The regularExpression is any expression that evaluates to a regular expression.

The optional positionVarsList consists of an even number of existing variables, separated by commas.

Value:

The matchChunk function returns true or false.

Comments:

If the regularExpression includes a pair of parentheses, the position of the substring matching the part of the regular expression inside the parentheses is placed in the variables in the positionVarsList. The number of the first character in the matching substring is placed in the first variable in the positionVarsList, and the number of the last character is placed in the second variable. Additional

starting and ending positions, matching additional parenthetical expressions, are placed in additional pairs of variables in the positionVarsList. If the matchChunk function returns false, the values of the variables in the positionVarsList are not changed.

The string and regularExpression are always case-sensitive, regardless of the setting of the caseSensitive property. (If you need to make a case-insensitive comparison, use "(?i)" at the start of the regularExpression to make the match case-insensitive.)

Important! Variables in the positionVarsList must be created before the matchChunk function is called. The matchChunk function, unlike the put command, does not create them automatically.

The matchChunk and matchText functions return the same value, given the same string and regularExpression. The difference between the two is that the matchChunk function records the positions of matched substrings in the optional positionVarsList variables, while the matchText function records the substrings themselves.

Tip: Revolution implements regular expressions compatible with the PCRE library. For detailed information about regular expression elements you can use with this function, see the PCRE manual at <<http://www.pcre.org/man.txt>>.

matchText

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

filter command, format function, matchChunk function, offset function, wholeMatches property, Regular Expressions Syntax Reference

Summary

Returns true if a regular expression is found in the specified string, false otherwise.

Syntax

matchText(string,regularExpression[,foundTextVarsList])

Example Code

```
matchText("Goodbye","bye") -- returns true
matchText("Goodbye","^Good") -- also returns true
matchText(phoneNumber,"([0-9]+)-([0-9]+-[0-9]+)","areaCode,phone")
```

Comments

Use the matchText function to check whether a string contains a specified pattern.

Parameters:

The string is any expression that evaluates to a string.

The regularExpression is any expression that evaluates to a regular expression.

The optional foundTextVarsList consists of one or more names of existing variables, separated by commas.

Value:

The matchText function returns true or false.

Comments:

If the regularExpression includes a pair of parentheses, the substring matching the part of the regular expression inside the parentheses is placed in the first variable in the foundTextVarsList. Additional substrings, matching additional parenthetical expressions within the regularExpression, are placed in

additional variables in the foundTextVarsList. The number of parenthetical expressions in the regularExpression should match the number of variables in the foundTextVarsList.

If the matchText function returns false, the values of the variables in the foundTextVarsList are not changed.

For example, the following matchText function call extracts the user name and email address from a typical email "From" line:

```
matchText(myVar,"^From: (.*) <(.+@.+)>",userName,userAddress)
```

There are two parenthetical expressions in the regularExpression above: "(.*)" and "(.+@.+)". If the function returns true—that is, if the string in myVar matches the regular expression—then the substring of myVar that matches the first of these parenthetical expressions is placed in the variable called userName; the second is placed in the variable userAddress.

The string and regularExpression are always case-sensitive, regardless of the setting of the caseSensitive property. (If you need to make a case-insensitive comparison, use "(?i)" at the start of the regularExpression to make the match case-insensitive.)

Important! Variables in the foundTextVarsList must be created before the matchText function is called. The matchText function, unlike the put command, does not create them automatically.

The matchText and matchChunk functions return the same value, given the same string and regularExpression. The difference between the two is that the matchText function records the text of matched substrings in the optional foundTextVarsList, which the matchChunk function records the character positions of the matched substrings.

Tip: Revolution implements regular expressions compatible with the PCRE library. For detailed information about regular expression elements you can use with this function, see the PCRE manual at [<http://www.pcre.org/man.txt>](http://www.pcre.org/man.txt).

matrixMultiply

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

* operator, extents function, keys function, multiply command, transpose function

Summary

Returns the matrix product of two arrays.

Syntax

`matrixMultiply(firstArray,secondArray)`

Example Code

```
put matrixMultiply(currentLevels,levelAdjustments)
```

Comments

Use the matrixMultiply function to perform matrix multiplication.

Parameters:

The firstArray and secondArray are both two-dimensional array variables whose elements are numbers, and whose keys are sequential numbers.

Value:

The matrixMultiply function returns an array of numbers.

Comments:

A two-dimensional array is an array whose elements have a two-part key to describe them. You can visualize such an array as a set of rows and columns: the first part of each element's key is the row number, and the second part is the column number. For example, the expression `myArray[3,2]` describes the element of myArray which is in the third row, second column.

The matrix product of two arrays is itself an array. Each element `myArray[M,N]` of the product is obtained by multiplying each element of the Mth row by the corresponding element of the Nth column. The resulting numbers are added together to obtain the element in the Mth row and Nth column.

The number of rows in the firstArray must be the same as the number of columns in the secondArray, and the number of rows in the secondArray must match the number of columns in the firstArray. However, the number of rows in the firstArray and secondArray need not be the same.

max

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

> operator, >= operator, average function, min function, sum function, Recipe for setting the red channel of an object

Summary

Returns the largest number from a list of numbers.

Syntax

max(numbersList)

Example Code

```
max(17,34,8,1) -- returns 34  
put max(returnedValues) into upperLimit
```

Comments

Use the max function to find the highest value in a set of values.

Parameters:

The numbersList is a comma-separated list of numbers, or an expression that evaluates to such a list, or an array containing only numbers.

Value:

The max function returns a number.

Comments:

You can use the max and min functions together to limit a value to a certain range. For example, the expression

```
max(10,min(myValue,100))
```

yields a number between 10 and 100. If myValue is within the limits, the expression is equal to myValue; if it is greater than 100, the expression evaluates to 100; and if it is less than 10, the expression evaluates to 10.

If the numbersList is empty, the max function returns zero.

Changes to Transcript:

The ability to use an array was introduced in version 1.1. In previous versions, only lists of numbers could be used with the max function.

maxHeight

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

height property, maxWidth property, minHeight property, resizable property, resizeStack message, windowBoundingRect property, How to prevent the user from resizing a window

Summary

Specifies how tall a stack window can be made when it's being resized.

Syntax

set the maxHeight of stack to pixels

Example Code

```
set the maxHeight of stack "Help" to 500
set the maxHeight of the target to (item 2 of the screenRect - 48)
```

Comments

Use the maxHeight property if you want the user to be able to resize the stack window, but limited to a height you choose.

Value:

The maxHeight of a stack is a positive integer.

By default, the maxHeight property of newly created stacks is set to 65535.

Comments:

The maxHeight is the maximum height in pixels. The height does not include the window's title bar or borders.

The maxHeight property does not prevent a handler from changing the stack's height property (or related properties such as the rectangle). It affects only user actions. If you set the stack's height to a value greater than the maxHeight, it is reset to the maxHeight when you close and reopen the stack.

If the stack's resizable property is false, the setting of this property has no effect.

maximize

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

decorations property, minimize keyword, title keyword, zoomBox property

Summary

Used with the decorations property to indicate whether the maximize or zoom box appears in a window's title bar.

Syntax

Example Code

```
set the decorations of this stack to maximize
```

Comments

Use the maximize keyword to turn a window's maximize or zoom box on or off.

Comments:

Setting a stack's decorations property to maximize automatically includes title in the value of the decorations, turning on the window's title bar.

Cross-platform note: On Windows systems, the menu decoration must be set along with maximize: you cannot use maximize without including menu.

maxWidth

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

maxHeight property, minWidth property, resizable property, resizeStack message, width property, windowBoundingRect property, How to prevent the user from resizing a window

Summary

Specifies how wide a stack window can be made when it's resized.

Syntax

set the maxWidth of stack to pixels

Example Code

```
set the maxWidth of the defaultStack to 300
set the maxWidth of stack "Prefs" to the height of stack "Prefs"
```

Comments

Use the maxWidth property if you want the user to be able to resize the stack window, but no larger than a width you choose.

Value:

The maxWidth of a stack is a positive integer.

By default, the maxWidth property of newly created stacks is set to 65535.

Comments:

The maxWidth is the maximum width in pixels. The width does not include the window's borders.

The maxWidth property does not prevent a handler from changing the stack's width property (or related properties such as the rectangle). It affects only user actions. If you set the stack's width to a value greater than the maxWidth, it is reset to the maxWidth when you close and reopen the stack.

If the stack's resizable property is false, the setting of this property has no effect.

mcEncrypt

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

ask password command, password property

Summary

Reserved for internal use.

Syntax

Example Code

Comments

MCISendString

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

lowResolutionTimers property, platform function, play command, queryRegistry function, setRegistry function

Summary

Sends a command to the Media Control Interface system, and returns the result of the command.

Syntax

MCISendString(MCICommand)

Example Code

```
MCISendString("capability overlay can freeze")  
MCISendString(field "Media Commands")
```

Comments

Use the MCISendString function on Windows systems to control multimedia devices.

Parameters:

The MCICommand is a string containing an MCI command.

Value:

The MCISendString function returns the value the device sends back.

Comments:

The Media Control Interface is a Microsoft standard that enables Windows systems to communicate with digitizers, laser-disc players, TV cards, and other multimedia devices.

If the device sends back an error message, the result is set to that message. If the command was successful, the result is set to empty.

For information about the MCI commands that can be used with a device, contact the manufacturer of the device.

mcLicense

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

licensed function

Summary

Reserved for internal use.

Syntax

Example Code

Comments

md5Digest

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

base64Decode function, base64Encode function, charToNum function

Summary

Returns a 128-bit value based on a string.

Syntax

the md5Digest of dataString
md5Digest(dataString)

Example Code

```
md5Digest("ABC")  
set the savedPrint of field 1 to the md5Digest of field 1  
if the savedPrint of field 1 is not the md5Digest of field 1 then beep
```

Comments

Use the md5Digest function to create a fingerprint of the dataString which can be checked later to ensure that the string has not changed.

Parameters:

The dataString is any string, or an expression that evaluates to a string.

Value:

The md5Digest function returns a 128-bit value. (If you view this value as a string, it is 16 characters long.)

Comments:

The value returned for one dataString is mathematically unlikely to be identical to the value returned for an altered version of the dataString. Therefore, you can check the md5Digest of a string against a saved md5Digest to determine whether the string has changed since you saved the value.

For technical information about the MD-5 digest algorithm, see RFC 1321 at
<<http://www.ietf.org/rfc/rfc1321.txt>>.

me

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

it keyword, target function, target keyword, text property, Recipe for a Find field, Recipe for a scrolling text banner

Summary

Equivalent to the object that contains the currently running handler.

Syntax

me

Example Code

```
set the borderWidth of me to it
put the short ID of me into savedID
put me into myVariable -- puts contents, if me is a container
put me into myVariable -- puts name, if me is not a container
```

Comments

Use the me keyword within a handler to determine which object's script is executing.

Comments:

The me keyword is a reference to the object whose script contains the current handler.

If the currently executing handler is in the script of the object that received the original message, then me is the same as the object whose name is returned by the target function. For example, suppose a button script contains a mouseDown handler. The value of the target function within that mouseDown handler is the same as the name of me: the name of the button.

However, if the mouseDown handler is in the card's script instead of the button's, me is not the same as the object specified by the target. In this case, me is the card, but the target function returns the button's name, because the button is the object that first received the mouseDown message.

The me keyword can be used anywhere an object reference is used.

median

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

average function, max function, min function, round function, standardDeviation function, sum function

Summary

Returns the median (middle number) of a list of numbers.

Syntax

median(numbersList)

Example Code

```
median(6,22,8) -- returns 8  
put median(incomeLevels) into field "2001 Median Income"
```

Comments

Use the median function to find the value that best represents a group of values.

Parameters:

The numbersList is a comma-separated list of numbers, or an expression that evaluates to such a list, or an array containing only numbers.

Value:

The median function returns a number.

Comments:

If the number of items in the numbersList is odd, the median function returns the middle value of the list when it is sorted in numeric order. If the number of items is even, the median function takes the two middle values and averages them to produce the median.

If the list consists of fewer than three numbers, or if the numbers are evenly distributed, the median is equal to the average.

mediaTypes

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

enabledTracks property, trackCount property, tracks property

Summary

Specifies one or more types of media used in in a QuickTime movie.

Syntax

get the mediaTypes of player

Example Code

```
if the mediaTypes of player 1 contains "audio" then increaseLoudness
```

Comments

Use the mediaTypes property to find out what kinds of media a QuickTime movie contains.

Value:

The mediaTypes is a list of one or more of the following, separated by commas:

- video
- audio
- text
- qtvr
- sprite
- flash

This property is read-only and cannot be set.

Comments:

Most QuickTime movies provide more than one media type. For example, a typical movie might provide one video and one audio track.

If more than one track in the movie uses the same media type, the media type is specified only once in the mediaTypes property. For example, "audio" appears only once in the mediaTypes, even if the movie contains more than one audio track.

menu

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

decorations property, disable menu command, doMenu command, enable menu command, menuItem keyword, About menus and the menu bar

Summary

Used with the decorations property to specify that a window displays a menu bar. When used as the style property of a button, specifies that the contents of the button is displayed as a menu. Also used to designate a menu by name or number.

Syntax

Example Code

```
disable menu "Text"  
set the decorations of stack "Help" to "menu,maximize"
```

Comments

Use the menu keyword to give a window a menu bar (on Unix and Windows systems).

Comments:

If a button's style is set to "menu", its menuMode property determines how the menu is displayed when the user clicks the button.

You can use the menu keyword to refer to one of the buttons in the current menu bar. For example, if the first menu in the menu bar is called "File", the expression the name of menu 1 evaluates to "button "File"". (Remember that in Revolution, menus are usually implemented as buttons: the button's style and menuMode properties control how it is displayed, and the button's text property is used as the list of menu items.)

Cross-platform note: The menu decoration has no effect on Mac OS and OS X systems. On Windows systems, the menu decoration must be set along with the maximize or minimize decorations: you cannot use maximize or minimize without including menu.

menubar

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

defaultMenubar property, editMenus property, lockMenus property, About menus and the menu bar, How to switch between menu bars, Tools menu > Menu Builder

Summary

Specifies the name of a menu bar to use when a stack is frontmost.

Syntax

set the menubar of stack to {groupname | empty}

Example Code

```
set the menubar of this stack to "Menus"
```

Comments

Use the menubar property to specify which menus appear in the menu bar on Mac OS systems when a stack is the active window.

Value:

The menubar of a stack is the name of a group.

By default, the menubar of newly created stacks is set to empty.

Comments:

On Mac OS systems, the menu bar appears at the top of the screen. On Unix and Windows systems, the menu bar appears at the top of the stack window. (Revolution menus are created with buttons. Each menu is a button whose menuMode property is set to "pulldown"; these buttons are then grouped to form a menu bar.)

The menubar is the group that contains the buttons used to build the menu bar. This menu bar is used when the stack window is active, replacing the defaultMenubar. If the menubar of a stack is empty, the stack does not have its own custom menu bar, and the defaultMenubar is used when the stack is active.

On Mac OS systems, when a stack's menubar property is set, the stack is scrolled and resized on Mac OS systems so that the group is not visible in the stack window. (On Unix and Windows systems, this is not necessary, since the menu bar is normally displayed in the window.) To scroll the stack window back down so you can see and select the group's objects, set the editMenus property to true.

menuButton

function

Synonyms

menuObject

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

me keyword, option command, popup command, pulldown command, selectedObject function, target function

Summary

Returns the name of the button that triggered display of a stack menu.

Syntax

the menuButton

menuButton()

Example Code

```
the menuButton  
hide the menuButton
```

Comments

Use the menuButton function within a handler in a stack menu to determine which button opened the menu.

Value:

The menuButton function returns the name property of the button.

You create a stack menu by laying out the menu items as buttons in a stack window, then setting the menuName property of a button in another stack to the menu stack's name. Clicking the button displays the menu stack as a menu. If you use the same stack menu as the menuName of more than one button, the menuButton tells you which button the user clicked to display the stack menu.

For example, suppose you create a stack menu listing all your mailboxes, and display it when clicking both a "Delete" button and an "Open" button. Checking the menuButton from within the stack menu lets you respond appropriately to the user's choosing a mailbox, either deleting it or opening it.

menuHistory

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

menuButton function, menuMode property, menuName property, menuPick message, option command, popup command, pulldown command, selectedLine function, How to change the selected item in an option menu or combo box, How to determine the current selection in a menu, How to switch between button tabs, Recipe for a Window menu, Recipe for checkmarking a menu item

Summary

Specifies the number of the currently selected item of the menu that belongs to a button.

Syntax

set the menuHistory of button to itemNumber

Example Code

```
set the menuHistory of button "Tabs" to 1
```

Comments

Use the menuHistory property to change the selected item in a menu, or to find out which menu item is currently selected.

Value:

The menuHistory of a button is an integer between 1 and the number of menu items in the menu.

By default, the menuHistory property of newly created buttons is set to 1.

Comments:

When you set the menuHistory property, a menuPick message is sent to the button.

If the button's menuMode is "comboBox", setting its menuHistory also changes the button's label to the new menu item.

If the button's menuMode is "tabbed", setting its menuHistory also changes the active tab.

If the button's `menuMode` is "option", setting its `menuHistory` changes the label. It also determines which menu item is under the mouse pointer when the menu next appears. Make sure to set the `menuHistory` property of an option menu whenever you change the current choice, so that the choice is under the mouse pointer when the user clicks the menu.

Note: The effect of the `menuHistory` property in cascading menus is ambiguous. Avoid setting or relying on the `menuHistory` of a cascading menu.

menuItem

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

disable menu command, doMenu command, enable menu command, menu keyword

Summary

Used with the disable menu, doMenu, enable menu, and select commands, to designate a menu item in a menu.

Syntax

Example Code

```
enable menuItem 3 of button "Externals"  
select menuItem 2 of menu "File"
```

Comments

Use the menuItem keyword to enable or disable a menu item, or to perform a menu action.

Comments:

Unlike menus, menu items must be specified by number. The topmost item in a menu is numbered 1.

menuLines

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

menuMode property, option command

Summary

Specifies the number of visible lines in a drop-down list menu belonging to a button.

Syntax

set the menuLines of button to number

Example Code

```
set the menuLines of button "Options" to 10
```

Comments

Use the menuLines property to control the appearance of combo box or option menus.

Value:

The menuLines of a button is a positive integer.

By default, the menuLines property of newly created buttons is set to 5.

Comments:

When a menu containing a drop-down list opens, it displays a fixed number of lines (with a scrollbar to access any additional lines). The menuLines property specifies how many lines are displayed at a time.

This property affects only buttons whose menuMode property is set to "option" (on Unix and Windows systems only) or "comboBox". Other menu types are not affected by the setting of the menuLines.

If the button's style property is not menu, the setting of the menuLines property has no effect.

menuMessage

property

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

menuItem keyword, send command

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the menuMessage of menuItem to messageName

Example Code

Comments

In HyperCard, the menuMessage property determines what message is sent when the user chooses a menu item.

A handler can set the menuMessage to any value without causing a script error, but the actual action of the menu is not changed.

menuMode

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cascade keyword, comboBox keyword, menubar property, menuName property, menuPick message, option command, option keyword, popup command, popup keyword, pullDown command, pulldown keyword, style property, tabbed keyword, How to switch between button tabs

Summary

Determines the appearance and behavior of menus associated with a button.

Syntax

set the menuMode of button to menuType

Example Code

```
set the menuMode of button "Edit" to pulldown
```

Comments

Use the menuMode property to specify the appearance and behavior of the menu associated with a button.

Value:

The menuMode of a button is one of pullDown, cascade, popup, tabbed, comboBox, or option.

Comments:

Any button's contents can be viewed as a menu if the button's style property is set to "menu". The menuMode setting creates menus of different types from the button's contents. (You set the button's contents with the text property.)

You can also associate a stack with the button, using the menuName property. In this case, the contents of the stack's first card are displayed as a menu.

pulldown: Displays each line of the button's contents as a menu item in a normal pulldown menu. Use this menuMode for buttons that are grouped into a menubar.

Setting a button's `menuMode` to "pulldown" has the same result as creating the equivalent menu items (as buttons) in a stack, then using the pulldown command to display the stack as a menu.

Note: On Mac OS and OS X systems, pulldown menus in a window are drawn by the standard operating system routines if the button's `showBorder` property is set to true and its `borderWidth` is not zero. Pulldown menus in the menu bar are always drawn by the operating system.

cascade: Displays each line of the button's contents as a menu item in a hierarchical menu. (The button must be part of a stack menu.)

popup: Displays each line of the button's contents as a menu item in a popup menu. The menu appears at the point of the mouse click.

Setting a button's `menuMode` to "popup" has the same result as creating the equivalent menu items (as buttons) in a stack, then using the popup command to display the stack as a menu.

tabbed: Displays the button's contents as a horizontal list of tabs, like the tabs on file folders. Each line of the button's contents is the title of a tab. Usually, tabbed buttons are used for dialog boxes or palettes containing a different group of controls for each tab. You can handle the `menuPick` message to display the correct group when a tab is clicked:

```
on menuPick newTab,oldTab -- sent when user clicks a tab
    lock screen -- hide the swap
    hide group oldTab
    show group newTab
    unlock screen
end menuPick
```

comboBox: Displays the button's contents as a drop-down scrolling list, with an editable field at the top.

Note: If a button's `menuMode` is set to "comboBox", the button receives field messages. For example, when the user clicks in the editable field, an `openField` message is sent to the button.

option: Displays an option menu (when the `lookAndFeel` property is set to "Motif"), a drop-down list (when the `lookAndFeel` property is set to "Windows 95"), or a Mac-style popup menu (when the `lookAndFeel` property is set to "Appearance Manager" or "Macintosh"). Setting a button's `menuMode` to option has the same result as creating the equivalent menu items (as buttons) in a stack, then using the option command to display the stack as a menu.

Regardless of the `menuType`, a `menuPick` message is sent to the button when the user chooses a menu item from the menu.

If the button's `style` property is not set to "menu", the setting of its `menuMode` property has no effect.

menuMouseButton

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

menuButton function, menuMode property, menuName property, option command, popup command, pulldown command

Summary

Specifies which mouse button can be used to access the menu associated with a button.

Syntax

set the menuMouseButton of button to {zero | buttonNumber}

Example Code

```
set the menuMouseButton to 3
```

Comments

Use the menuMouseButton property to limit a menu so it can only be accessed with a specific mouse button.

Value:

The menuMouseButton is an integer between zero and 3.

By default, the menuMouseButton property of newly created buttons is set to zero.

Comments:

If the menuMouseButton is zero, any mouse button can be used to open the menu. If the menuMouseButton is a number from 1 to 3, only that mouse button opens the menu.

The mouseButtonNumber specifies which mouse button was pressed:

• 1 is the mouse button on Mac OS systems and the left button on Windows and Unix systems.

• 2 is the middle button on Unix systems.

• 3 is the right button (on Windows and Unix systems) or Control-click (on Mac OS and OS X systems).

menuName

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

menubar property, menuMode property, menuPick message, option command, popup command, pulldown command

Summary

Specifies the stack where the menu associated with a button is located.

Syntax

set the menuName of button to stack

Example Code

```
set the menuName of card button "Popup" to stack "Items"
```

Comments

Use the menuName property to associate a stack menu with a button.

Value:

The menuName of a button is a stack reference.

By default, the menuName property of newly created buttons is set to empty.

Comments:

When the button is clicked, the specified stack is opened as a menu. The appearance and behavior of the menu depend on the setting of the button's menuMode property.

If the menuName of a button is empty, its text is used as the contents of the menu.

If the button's style property is not menu, the setting of this property has no effect.

menuPick

message

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

doMenu command, menuHistory property, menuMode property, option command, popup command, pulldown command, How to change the selected item in an option menu or combo box, How to respond to quitting an OS X application, Recipe for a Window menu, Recipe for checkmarking a menu item

Summary

Sent to a button when a menu item is chosen from the menu associated with that button.

Syntax

menuPick chosenItem[|submenuName|],previousTab

Example Code

```
on menuPick theItem -- in a pulldown menu
  if theItem is "Close" then close the defaultStack
  else if theItem is "Quit" then quit
end menuPick
```

Comments

Handle the menuPick message to do something when the user chooses a menu item from a button menu, or chooses a tab in a tabbed button.

Parameters:

The chosenItem is the text of the menu item the user chose. If the menu item is part of a submenu, the menu item text is followed by a vertical bar (|) and the submenu's name.

The previousTab is the text of the menu item that was selected before the user chose a new menu item. This parameter is included only if the button's menuMode property is set to "tabbed".

Comments:

The menuPick message is sent when the user clicks a tab in a tabbed button, when the user chooses a menu item from the menu associated with a button, or when a button's menuHistory property is set by a handler.

The menuPick message is sent every time a menu item is chosen, even if the menu is a type that retains its state (such as an option menu) and the chosen menu item has not changed. (However, no menuPick message is sent when an already-chosen tab in a tabbed button is clicked, since no menu is displayed.)

To get the item number of the currently chosen menu item, use the menuHistory property.

Note: The & and / characters can be used in a pulldown menu to create special effects. To be shown (instead of creating these effects), either character can be doubled. (For example, placing the line "This && That" in a menu item results in the menu item being displayed as "This & That", with one ampersand.) When the user chooses a menu item with these special characters, the chosenItem parameter is the text that appears in the menu as displayed, not the text of that line of the menu button.

menus function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

defaultMenubar property, menu keyword, menubar property

Summary

Does not return a meaningful value and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

the menus

menus()

Example Code

Comments

In HyperCard, the menus function returns a list of menus in the menubar.

In Revolution, the menus function always returns empty.

merge

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

return control structure, value function

Summary

Evaluates any expressions in a string, replaces them with their value, and returns the transformed string.

Syntax

`merge(stringToEvaluate)`

Example Code

```
merge("1+1 equals [[1+1]]") -- returns "1+1 equals 2"  
merge("The current folder is [[the defaultFolder]]")
```

Comments

Use the merge function to combine text with the result of evaluating expressions in a single string.

Parameters:

The `stringToEvaluate` is a string of any length, which may include expressions enclosed in double square brackets (`[[expression]]`) and return statements enclosed in "`<?`" and "`?>`" (`<?return expression?>`).

Value:

The merge function returns a string.

Comments:

The merge function evaluates any expressions in double square brackets, and replaces them with the expression's value.

It also executes any return statements enclosed in "`<?`" and "`?>`", and replaces them by the value returned.

message box

keyword

Synonyms

message window, message, msg box, msg window, msg

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

do command, put command, Tools menu > Message Box, Shortcut to clear the message box

Summary

Indicates the message box.

Syntax

Example Code

```
show message box  
put the result into message
```

Comments

Use the message box keyword to display information to the user.

Comments:

The message box is a container as well as a window. You can show and hide the message box, put information into it, and get information from it.

The two-word synonyms message window and msg window can be used only to show and hide the message box. To put a value into the message box or read the value in the message box, use the forms message, msg, message box, and msg box.

metaKey

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

altKey function, commandKey function, controlKey function, down constant, keyDown message, keysDown function, keyUp message, optionKey function, shiftKey function, up constant

Summary

Returns the state of the Meta key.

Syntax

the metaKey

metaKey()

Example Code

```
get metaKey()  
if the metaKey is down then go next card
```

Comments

Use the metaKey function to check whether the Meta key, Option key, or Alt key is being pressed. You can use metaKey to add alternative capabilities to user actions such as clicking.

Value:

The metaKey function returns down if the key is pressed and up if it's not.

Comments:

The altKey, optionKey, and metaKey functions all return the same value. Which one to use is a matter of preference.

The terminology varies depending on platform. Users of different operating systems may know this key as the Option key (Mac OS systems), Meta key (Unix systems), or Alt key (Windows systems).

metal

property

Synonyms

Objects

stack

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

backgroundPattern property, decorations property, How to get a striped background in OS X

Summary

Displays a window with a textured-metal appearance.

Syntax

set the metal of stack to {true | false}

Example Code

```
set the metal of this stack to false
set the metal of stack "Calculator" to true
```

Comments

Use the metal property to give OS X windows a distinctive appearance suitable for applications such as calculators, oscilloscopes, video camera controllers, and other uses that emulate or control a physical device

Value:

The metal property of a stack is true or false. By default, the metal of a newly created stack is set to false.

Comments:

Before using the metal property, consult Apple's Aqua user-interface guidelines to determine whether the use is appropriate. In general, metal windows should be used to represent physical devices or interfaces to such devices, and should be used only for the application's main window (not for palettes and dialog boxes).

On Mac OS, Unix, and Windows systems, the metal property has no effect.

Note: The setting of this property determines whether the stack's decorations property includes "metal".

middle

keyword

Synonyms

mid

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

any keyword, first keyword, last keyword, number property

Summary

Designates the middle member of a set.

Syntax

Example Code

```
go to middle card -- halfway through the stack
put the middle char of myInput into keyVar
select the middle line of field 22
```

Comments

Use the middle keyword in an object reference or chunk expression.

Comments:

The middle keyword can be used to specify any object. It can also be used in a chunk expression to designate the middle chunk of the specified type.

If the set has an odd number of members, the middle keyword specifies the member halfway between beginning and end. For example,

the middle char of "abc"
is b.

If the set has an even number of members, the middle keyword specifies the first member of the last half of the set. For example,

the middle char of "abcd"
is c.

The word the is optional when using the middle keyword.

A reference to the middle of a string, with no chunk type specified, yields the entire string.

milliseconds

function

Synonyms

millisecond, millisec, millisecs

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

convert command, date function, milliseconds keyword, seconds function, ticks function, time function

Summary

Returns the number of milliseconds since the start of the eon.

Syntax

the milliseconds

milliseconds()

Example Code

```
the milliseconds
if the milliseconds - startMilliseconds < 10 then answer "Success!"
```

Comments

Use the milliseconds function to time events that must be checked more often than once per second.

Value:

The milliseconds function returns a positive integer.

Comments:

The milliseconds function returns the total number of milliseconds since midnight GMT, January 1, 1970.

A millisecond is one-thousandth of a second. If you don't need to time events with this much precision, use the seconds function or the ticks function.

milliseconds

keyword

Synonyms

millisecond,millisecs,millisec

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

milliseconds function, seconds keyword, ticks keyword

Summary

Designates the number of milliseconds in a time period.

Syntax

Example Code

```
send "choose browse tool" to me in 20 milliseconds -- 1/50th of a second
```

Comments

Use the milliseconds keyword to designate a time period with the wait or send commands.

Comments:

When used with the wait or send commands, the milliseconds keyword designates a time period measured in thousandths of a second.

mimeText

property

Synonyms

Objects

field

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

htmlText property

Summary

The mimeText property is not implemented and is reserved.

Syntax

Example Code

Comments

min

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

< operator, <= operator, average function, max function, sum function, Recipe for setting the red channel of an object

Summary

Returns the smallest number of a list of numbers.

Syntax

min(numbersList)

Example Code

```
min(12,3,8,7) -- returns 3  
min(0,8,-436) -- returns -436
```

Comments

Use the min function to find the lowest value in a group of values.

Parameters:

The numbersList is a comma-separated list of numbers, or an expression that evaluates to such a list, or an array containing only numbers.

Value:

The min function returns a number.

Comments:

You can use the min and max functions together to limit a value to a certain range. For example, the expression

```
min(1,max(myValue,0))
```

yields a number between zero and 1. If myValue is within the limits, the expression is equal to myValue; if it is greater than 1, the expression evaluates to 1; and if it is less than zero, the expression evaluates to zero.

If the numbersList is empty, the min function returns zero.

Changes to Transcript:

The ability to use an array was introduced in version 1.1. In previous versions, only lists of numbers could be used with the min function.

minHeight

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

height property, maxHeight property, minWidth property, resizable property, resizeStack message, How to prevent the user from resizing a window

Summary

Specifies how short a stack window can be made when it's resized.

Syntax

set the minHeight of stack to pixels

Example Code

```
set the minHeight of this stack to the height of this stack
```

Comments

Use the minHeight property if you want the user to be able to resize the stack window, but no smaller than a height you choose. For example, you might want to let the user make the stack smaller, but not so small that the stack's navigation buttons can't be displayed.

Value:

The minHeight of a stack is a positive integer.

By default, the minHeight property of newly created stacks is set to 32.

Comments:

The minHeight is the minimum height in pixels. The height does not include the window's title bar or borders.

The minHeight property does not prevent a handler from changing the stack's height property (or related properties such as the rectangle). It affects only user actions. If you set the stack's height to a value less than the minHeight, it is reset to the minHeight when you close and reopen the stack.

If the stack's resizable property is false, the setting of this property has no effect.

Cross-platform note: On Mac OS and OS X systems, if the current card has a menu bar and the `editMenus` property is false (that is, if the stack window is scrolled up so the menubar buttons are not visible in the window), the `minHeight` does not include the height of the menu bar. However, on Windows and Unix systems, the `minHeight` includes the height of the menu bar, since on these platforms the menu bar is in the stack window. This means that if you set a `minHeight` for a stack that contains a menu bar, you may need to adjust it depending on platform so that the `minHeight` on Unix and Windows systems includes the height of the menu bar, while the `minHeight` on Mac OS and OS X systems does not. (The standard height of menu bars created with the Menu Builder is 21 pixels.)

minimize

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

decorations property, maximize keyword, title keyword

Summary

Used with the decorations property to indicate whether the minimize button appears in a window's title bar.

Syntax

Example Code

```
set the decorations of stack "Help" to "minimize, menu"
```

Comments

Use the minimize keyword to turn a window's minimize button on or off.

Comments:

Setting a stack's decorations property to minimize automatically includes title in the value of the decorations, turning on the window's title bar.

On Windows systems, the menu decoration must be set along with minimize: you cannot use minimize without including menu.

minimizeBox

property

Synonyms

collapseBox

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeBox property, decorations property, iconic property, iconifyStack message, minimize keyword, unIconifyStack message, zoomBox property

Summary

Shows a window's minimize box or collapse box.

Syntax

set the minimizeBox of stack to {true | false}

Example Code

```
set the minimizeBox of me to true
set the minimizeBox of the templateStack to false
```

Comments

Use the minimizeBox property to display the collapse box or minimize box in a window's title bar.

Value:

The minimizeBox property of a stack is true or false. By default, the minimizeBox of a newly created stack is set to true.

Comments:

The terminology varies depending on platform. The minimizeBox property determines whether the collapse box (Mac OS systems), minimize box (OS X and Windows systems) or iconify box (Unix systems) appears in a stack window's title bar.

Note: On OS X systems, if the minimizeBox property is false, the minimize box is disabled rather than hidden.

The setting of this property affects the decorations property, and vice versa. Setting a stack's minimizeBox property determines whether its decorations property includes "minimize" (or is "default", for window styles that normally include a minimize box). Conversely, setting a stack's decorations

property sets its `minimizeBox` to true or false depending on whether the decorations includes "minimize" (or is "default").

Setting the `minimizeBox` property causes the stack window to flash briefly.

minWidth

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

maxWidth property, minHeight property, resizable property, resizeStack message, width property, How to prevent the user from resizing a window

Summary

Specifies how narrow a stack window can be made when it's resized.

Syntax

set the minWidth of stack to pixels

Example Code

```
set the minWidth of this stack to 200
```

Comments

Use the minWidth property if you want the user to be able to resize the stack window, but no smaller than a width you choose. For example, you might want to let the user make the stack smaller, but not so small that the stack's navigation buttons can't be displayed.

Value:

The minWidth of a stack is a positive integer.

By default, the minWidth property of newly created stacks is set to 32.

Comments:

The minWidth is the minimum width in pixels. The height does not include the window's title bar or borders.

The minWidth property does not prevent a handler from changing the stack's width property (or related properties such as the rectangle). It affects only user actions. If you set the stack's width to a value less than the minWidth, it is reset to the minWidth when you close and reopen the stack.

If the stack's resizable property is false, the setting of this property has no effect.

mnemonic

property

Synonyms

Objects

button

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

acceleratorKey property, acceleratorText property, length function, name property

Summary

Specifies which character of a button's name can be used with the Alt key to trigger the button.

Syntax

set the mnemonic of button to number

Example Code

```
set the mnemonic of button "Avant!" to 1 -- uses the first letter, A
```

Comments

Use the mnemonic property to provide a keyboard substitute for clicking a button on Windows systems.

Value:

The mnemonic of a button is a positive integer. Zero indicates that there is no shortcut provided.

By default, the mnemonic property of newly created buttons is set to zero.

Comments:

The mnemonic is any number between 1 and the number of characters in the button's name. The character at that position is underlined, and pressing the Alt key with that character sends a mouseUp message to the button. If the number is zero, the button does not have an Alt key equivalent.

This property has no effect on Mac OS or Unix systems.

mod

operator

Synonyms

Objects

numeric

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

/ operator, div operator, round function, trunc function, Operator Precedence Reference

Summary

Evaluates to the remainder left over when one number is divided by another.

Syntax

number mod divisor

Example Code

```
23 mod 5 -- evaluates to 3 (23 div 5 is 4, with 3 left over)
12 mod 4 -- evaluates to zero
23 mod -5 -- evaluates to 3
```

Comments

Use the mod operator to perform modulus arithmetic.

Parameters:

The number is a number, or an expression that evaluates to a number, or an array containing only numbers.

The divisor is any non-zero number. If the number is an array, the divisor is either a non-zero number or an array containing only non-zero numbers.

Comments:

If the number to be divided is an array, each of the array elements must be a number. If an array is divided by a number, each element is divided by the number. If an array is divided by an array, both arrays must have the same number of elements and the same dimension, and each element in one array is divided by the corresponding element of the other array.

If an element of an array is empty, the mod operator treats its contents as zero.

If number can be divided evenly into divisor, the expression number mod divisor is zero.

Attempting to divide by zero causes an execution error.

Note: Using non-integer number and divisor usually produces sensible results. However, mathematically, modulus is generally defined as a function over the integers, and the results using non-integers may not consistently be what you expect.

Changes to Transcript:

The option to divide arrays was introduced in version 1.1. In previous versions, only single numbers could be used with the mod operator.

modal command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer command, ask command, dialogData property, go command, mode property, modeless command, palette command, sheet command, style property, topLevel command, About windows, palettes, and dialog boxes, How to close a modal dialog box, How to create a custom dialog box, How to display a dialog box, Why does a stack window open in the wrong mode?

Summary

Opens a stack as a modal dialog box.

Syntax

modal stack

Example Code

```
modal stack "Custom Answer Dialog"  
modal stack x of stacksToPresent
```

Comments

Use the modal command to display a stack as a custom modal dialog box.

Parameters:

The stack is any stack reference.

Comments:

The modal command opens the stack centered over the defaultStack, regardless of the stack's rectangle and location properties.

While a modal dialog box is open, other windows cannot be edited or brought to the front. Because of this, you should use modal dialog boxes only when a stack must obtain feedback from the user before it can continue.

If the stack is already open, the modal command closes the stack and reopens it as a modal dialog box, so closeStack and openStack, closeCard and openCard, and (if applicable) closeBackground and openBackground messages are sent to the current card as a result of executing this command. Use the

lock messages command before executing modal if you want to prevent the close messages from being sent; the open messages are sent regardless of the setting of the lockMessages property.

If the stack is already displayed as a modal dialog box, the modal command does not close and reopen it.

The modal command pauses the running handler until the modal dialog box is dismissed (usually by clicking a button in the modal dialog box). To return information to the handler about which button was clicked, in the button's script, set a global variable or custom property. After the dialog box is dismissed, the handler can query this variable or property and act accordingly.

Modal dialog boxes cannot be resized or edited. To edit a modal dialog box, use the topLevel command to display it in an editable window. The Browse tool is used in modal dialog boxes, regardless of the current setting of the tool property.

mode

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantModify property, defaultStack property, iconic property, menuMode property, modal command, modeless command, palette command, style property, topLevel command, topStack function, Why can't I display a stack as a normal window?, Why does a stack window open in the wrong mode?

Summary

Reports the type of window a stack is displayed in.

Syntax

get the mode of stack

Example Code

```
if the mode of this stack > 2 then topLevel this stack
```

Comments

Use the mode property to determine how the user can interact with the stack.

Value:

The mode of a stack is an integer between zero and 14.

This property is read-only and cannot be set.

Comments:

The mode of a stack is one of the following numbers:

0: closed but loaded

The stack window is closed, but the stack is still in memory. A stack can have this mode if its destroyStack property is set to false and the stack has been opened, then closed.

1: editable window

The stack is open and displayed in an editable window. A stack has this mode if its style is "topLevel" or if it was opened with the topLevel command.

2: non-editable window

The stack is open and displayed in a normal window, but is not editable. A stack has this mode if its style is "topLevel" or if it was opened with the topLevel command, and the stack's cantModify property is set to true.

3: modeless dialog

The stack is open and displayed as a modeless dialog box. A stack has this mode if its style is "modeless" or if it was opened with the modeless command

4: palette

The stack is open and displayed as a palette. A stack has this mode if its style is "palette" or if it was opened with the palette command.

5: modal dialog

The stack is open and displayed as a modal dialog box. A stack has this mode if its style is "modal" or if it was opened with the modal command.

6: sheet

The stack is open and displayed as a sheet. A stack has this mode if it was opened with the sheet command. (If the sheet command was used on a platform other than OS X, the stack is displayed as a modal dialog box instead of a sheet. In this case, its mode is 5.)

7: pulldown stack menu

The stack is open and displayed as a pulldown menu. A stack has this mode if it was opened with the pulldown command, or if it is the menuStack of a button whose style is set to "menu" and whose menuMode is set to pulldown.

8: popup stack menu

The stack is open and displayed as a popup menu. A stack has this mode if it was opened with the popup command, or if it is the menuStack of a button whose style is set to "menu" and whose menuMode is set to "popup".

9: option stack menu

The stack is open and displayed as an option menu. A stack has this mode if it is the menuStack of a button whose style is set to "menu" and whose menuMode is set to "option".

10: submenu in a stack menu

The stack is open and displayed as a cascading menu. A stack has this mode if it is the menuStack of a button whose style is set to "menu" and whose menuMode is set to "cascade".

11: combo box stack menu

The stack is open and displayed as a combo box. A stack has this mode if it is the menuStack of a button whose style is set to "menu" and whose menuMode is set to "comboBox".

12: collapsed

The stack is open but has been collapsed (Mac OS), iconified (Unix), or minimized (Windows).

13: drawer

The stack is open as a drawer beside another window. A stack has this mode if it was opened with the drawer command.

If two stacks with different modes are open, the stack whose mode property is lower takes precedence when determining which stack is the topStack. This means, for example, that menu items (such as Object menu Stack Properties) that act on the current stack may not be able to operate correctly with a stack whose cantModify is set to true (and whose mode is therefore 2) as long as another, modifiable stack (mode equal to 1) is open.

modeless command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantSelect property, dialogData property, go command, modal command, mode property, palette command, sheet command, style property, topLevel command, About windows, palettes, and dialog boxes, How to create a custom dialog box, Why does a stack window open in the wrong mode?

Summary

Opens a stack in a modeless dialog window.

Syntax

modeless stack

Example Code

```
modeless stack "Hello World"  
modeless the defaultStack
```

Comments

Use the modeless command to display a stack as a custom modeless dialog box.

Parameters:

The stack is any stack reference.

Comments:

A modeless dialog box behaves like an ordinary window, except that it cannot be edited. Use modeless dialog boxes to ask the user for information.

To edit a modeless dialog box, use the topLevel command to display it in an editable window.

The modeless command closes the stack and reopens it as a modeless dialog box, so closeStack and openStack, closeCard and openCard, and (if applicable) closeBackground and openBackground messages are sent to the current card as a result of executing this command. Use the lock messages command before executing modeless if you want to prevent the close messages from being sent; the open messages are sent regardless of the setting of the lockMessages property.

If the stack is already displayed as a modeless dialog box, the modeless command does not close and reopen it.

The Browse tool is used in modeless dialog boxes, regardless of the current setting of the tool property.

modem:
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

close file command, COM1: keyword, LPT1: keyword, open file command, printer: keyword, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands to specify the modem port on Mac OS systems.

Syntax

Example Code

```
read from file "modem:" until end
```

Comments

Use the modem: keyword to communicate through the modem serial port.

Comments:

To set the port parameters (such as baud rate, handshaking, and parity), set the serialControlString property before opening the port with the open file command.

To read data from the modem port, use the read from file command, specifying the keyword modem: as the file to read from.

To write data to the modem port, use the write to file command.

To use the printer port on Mac OS systems, use the printer: keyword. (To use serial ports other than the built-in printer or modem port, use the open driver command.)

To use serial ports on Windows systems, use the COM1: through COM9: keywords.

monthNames

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

convert command, date function, dateFormat function, english keyword, internet keyword, system keyword, weekdayNames function

Summary

Returns a list of month names used by the date function.

Syntax

the [long | abbr[rev[iated]]] | short] [english|system|internet] monthNames

the [internet] [english | system] monthNames

monthNames()

Example Code

```
put the monthNames into listToDelete
get the long english monthNames
```

Comments

Use the monthNames function to let the user choose from a list of month names, or to display a list of months in the language of the user's system preferences.

Value:

The monthNames function returns a set of twelve month names, one per line.

The monthNames form returns the full name of each month, as used in the long date.

The long monthNames form returns the same result as the monthNames form.

The abbreviated monthNames form returns the first three letters of the name of each month, as used in the abbreviated date.

The short monthNames form returns the month number of each month, as used in the short date.

Comments:

For each form of the monthNames, if the useSystemDate property is set to true, or if you specify the system monthNames, the list of names is provided in the language specified by the user's system preferences. If the useSystemDate is false or you specify the english monthNames, the list of names is provided in English.

mouse

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

commandKey function, controlKey function, mouseDown message, mouseLoc function, mouseRelease message, mouseUp message, optionKey function, shiftKey function

Summary

Returns the state of a mouse button.

Syntax

the mouse [of buttonNumber]

mouse([buttonNumber])

Example Code

```
the mouse
mouse(3)
if mouse(defaultButton) is up then goBackToStart
```

Comments

Use the mouse function to check whether the user is pressing a mouse button.

Parameters:

The mouseButtonNumber specifies which mouse button to check:

• 1 is the mouse button on Mac OS systems and the left button on Windows and Unix systems.

• 2 is the middle button on Unix systems.

• 3 is the right button on Windows and Unix systems and Control-click on Mac OS systems.

Value:

The mouse function returns down if the key is pressed and up if it's not.

Comments:

If you don't specify a buttonNumber, the mouse function returns the state of mouse button 1.

mouseChar

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

charToNum function, clickChar function, lockText property, mouseCharChunk function, mouseChunk function, mouseLine function, mouseLoc function, mouseMove message, mouseStack function, mouseText function

Summary

Returns the character the mouse pointer is over.

Syntax

the mouseChar

mouseChar()

Example Code

```
the mouseChar
if the mouseChar is a number then goToFootnote
```

Comments

Use the mouseChar function within a handler to determine which character the mouse is hovering over, in order to take some action based on the current character.

Value:

The mouseChar function returns a character.

Comments:

The mouseChar function only returns characters in a field. If the mouse pointer is not over a field, the mouseChar function returns empty.

To find the position of the mouseChar within the field, use the mouseCharChunk function.

mouseCharChunk

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

charToNum function, clickCharChunk function, lockText property, mouseChar function, mouseChunk function, mouseControl function, mouseLoc function, mouseMove message, mouseStack function, mouseText function

Summary

Returns a chunk expression describing the location of the character under the mouse pointer.

Syntax

the mouseCharChunk

mouseCharChunk()

Example Code

```
the mouseCharChunk  
if the mouseChar is empty then select the mouseCharChunk
```

Comments

Use the mouseChar function within a handler to determine which character the mouse is hovering over, in order to take some action based on the current character.

Value:

The mouseCharChunk function returns a chunk expression of the form char charNumber to charNumber of field fieldNumber.

Comments:

The mouseCharChunk function only returns locations in a field. If the mouse pointer is not over a field, the mouseCharChunk function returns empty.

The first and second character numbers in the return value are always identical, unless the mouse is over a field but there is no text under it. In this case, the mouseCharChunk returns a chunk expression of the form char charNumber to charNumber - 1 of field fieldNumber, indicating the start of the mouseLine. For example, if the mouse is over an empty field, the mouseCharChunk returns char 1 to 0 of field fieldNumber.

To get the actual character clicked, use the `mouseChar` function.

mouseChunk

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickChunk function, foundChunk function, lockText property, mouseChar function, mouseCharChunk function, mouseControl function, mouseLoc function, mouseMove message, mouseStack function, mouseText function, selectedChunk function

Summary

Returns a chunk expression describing the location of the word or text group under the mouse pointer.

Syntax

the mouseChunk

mouseChunk()

Example Code

```
the mouseChunk
if the shiftKey is down then select the mouseChunk
```

Comments

Use the mouseChunk function within a handler to determine which word or text group the mouse is hovering over, in order to take some action based on the current character.

Value:

The mouseChunk function returns a chunk expression of the form char startChar to endChar of field fieldNumber.

Comments:

The mouseChunk function only returns locations in a field. If the mouse pointer is not over a field, the mouseChunk function returns empty.

The return value reports the word the mouse is over. The startChar is the first character of the word, and the endChar is the last character. If the textStyle of the text is "link", the return value specifies the entire text group.

To get the text of the word or text group, use the mouseText function.

Important! Words are defined a little differently by the mouseChunk function than the way they are used in chunk expressions. A word, for purposes of the mouseChunk, is any text delimited by spaces, tabs, returns, or punctuation. If the mouse pointer is over a punctuation character, only that character is returned.

mouseClick

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

click command, clickLoc function, mouse function, mouseDown message, mouseLoc function, mouseUp message

Summary

Returns true if the user has clicked since the event that caused the current handler to run (or since the last time the mouseClick was checked).

Syntax

the mouseClick
mouseClick()

Example Code

```
the mouseClick  
wait until the mouseClick
```

Comments

Use the mouseClick function to keep track of clicks, or to trigger some action within a handler when the user clicks.

Value:

The mouseClick function returns true or false.

Comments:

On systems with multi-button mice, the mouseClick function tracks mouse button 1. Clicking mouse button 2 or 3 does not affect the value of the mouseClick.

Clicks generated by the click command do not affect the mouseClick function. Only an actual click can cause the mouseClick to return true.

Checking the mouseClick function clears any previous clicks. If the user clicks, and then the mouseClick is called, it returns true. If the mouseClick is then called again (and the user hasn't clicked again meanwhile), the function returns false.

mouseColor

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

colors property, foregroundColor property, hotspot property, mouseLoc function, mouseStack function, Recipe for translating a color name to an RGB numeric triplet

Summary

Returns the color of the pixel that the mouse pointer is over.

Syntax

the mouseColor
mouseColor()

Example Code

```
the mouseColor  
set the foregroundColor of graphic 1 to the mouseColor
```

Comments

Use the mouseColor function to find out what color the mouse is over or to select a color from the screen.

Value:

The mouseColor function returns a color reference consisting of three integers between zero and 255, separated by commas.

Comments:

The mouseColor is the color of the single pixel under the mouse pointer's hot spot. The hot spot is the active point of the cursor. For example, the arrow cursor's hot spot is at its tip, and to select an object, you must click it with the arrow tip.

The value returned by the mouseColor consists of three comma-separated numbers between zero and 255, specifying the level of each of red, green, and blue.

mouseControl

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clickField function, layer property, mouseEnter message, mouseLeave message, mouseLoc function, mouseMove message, mouseStack function

Summary

Returns the number of the control the mouse pointer is over.

Syntax

the mouseControl
mouseControl()

Example Code

```
the mouseControl  
if the number of the mouseControl is 2 then set the cursor to hand
```

Comments

Use the mouseControl function to change the stack's behavior depending on what control the mouse is over, or to perform some action on the control the mouse is over.

Value:

The mouseControl function returns a string of the form control controlNumber.

Comments:

The control number returned in the mouseControl function is the same as the control's layer property.

If the mouse is not over a control, the mouseControl function returns empty.

If the mouse button is down, the mouseControl function returns the control that was clicked, even if the mouse has moved to another control.

If the mouse is over a graphic that is not selected, its interior is considered to be within the graphic only if the graphic's filled property is true. If the graphic's filled is false and the graphic is not selected, the mouseControl returns the graphic's number only if the mouse is over the graphic's border.

mouseDoubleClickDown

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

click command, clickLoc function, doubleClickDelta property, doubleClickInterval property, mouse function, mouseDoubleUp message, mouseDown message

Summary

Sent when the user double-clicks.

Syntax

mouseDoubleClickDown mouseButtonNumber

Example Code

```
on mouseDoubleClickDown myButton -- open a file in an another application
    if myButton is not 1 then pass mouseDoubleClickDown
    launch the label of the target with (field "Editor App")
end mouseDoubleClickDown
```

Comments

Handle the mouseDoubleClickDown message to perform an action when the user double-clicks.

Parameters:

The mouseButtonNumber specifies which mouse button was pressed:

- 1 is the mouse button on Mac OS systems and the left button on Windows and Unix systems.
- 2 is the middle button on Unix systems.
- 3 is the right button on Windows and Unix systems and Control-click on Mac OS systems.

Comments:

The mouseDoubleClickDown message is sent to the control that was double-clicked, or to the card if no control was under the mouse pointer.

The mouseDoubleClickDown message is sent only when the Browse tool is being used. If an unlocked field is clicked with mouse button 1 or 2, no mouseDoubleClickDown message is sent.

Important! If the user clicks a transparent pixel in an image, the mouseDoubleClickDown message is sent to the object behind the image, not to the image.

mouseDoubleUp

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

click command, clickLoc function, doubleClickDelta property, doubleClickInterval property, mouse function, mouseDoubleDown message, mouseUp message

Summary

Sent when the user double-clicks and the mouse button is released.

Syntax

mouseDoubleUp mouseButtonNumber

Example Code

```
on mouseDoubleUp theMouseButton -- show dialog on Control/Right-click
  if theMouseButton is "3" then showConfigDialog
  else pass mouseDoubleUp
end mouseDoubleUp
```

Comments

Handle the mouseDoubleUp message to perform an action when the user double-clicks.

Parameters:

The mouseButtonNumber specifies which mouse button was pressed:

• 1 is the mouse button on Mac OS systems and the left button on Windows and Unix systems.

• 2 is the middle button on Unix systems.

• 3 is the right button on Windows and Unix systems and Control-click on Mac OS systems.

Comments:

The mouseDoubleUp message is sent to the control that was double-clicked, or to the card if no control was under the mouse pointer.

If a tool other than the Browse tool is being used, no mouseDoubleUp message is sent. If an unlocked field is clicked with mouse button 1 or 2, no mouseDoubleUp message is sent.

The doubleClickInterval and doubleClickDelta properties determine what Revolution accepts as a double click.

Important! If the user clicks a transparent pixel in an image, the mouseDoubleUp message is sent to the object behind the image, not to the image.

mouseDown

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

click command, clickLoc function, dragStart message, lockText property, mouse function, mouseDoubleDown message, mouseRelease message, mouseUp message, How to find out which object the user clicked, How to simulate a mouse click, How to throw away unwanted mouse clicks

Summary

Sent when the user presses the mouse button.

Syntax

mouseDown mouseButtonNumber

Example Code

```
on mouseDown theButton -- show popup menu on Control/Right-click
  if theButton is 3 then popup button "Configure"
  else pass mouseDown
end mouseDow
```

Comments

Handle the mouseDown message to perform an action when the user presses the mouse button, before the mouse button is released.

Parameters:

The mouseButtonNumber specifies which mouse button was pressed:

• 1 is the mouse button on Mac OS systems and the left button on Windows and Unix systems.

• 2 is the middle button on Unix systems.

• 3 is the right button on Windows and Unix systems and Control-click on Mac OS systems.

Comments:

The mouseDown message is sent to the control that was clicked, or to the card if no control was under the mouse pointer.

If the Browse tool is being used, and you click an unlocked field with mouse button 1 or 2, no mouseDown message is sent. If you click with button 3, the mouseDown message is sent even though the field is not locked.

Important! If the user clicks a transparent pixel in an image, the mouseDown message is sent to the object behind the image, not to the image.

mouseDownInBackdrop

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backdrop property, mouseDown message, mouseUpInBackdrop message

Summary

Sent when the user presses the mouse button while the mouse pointer is in the backdrop..

Syntax

mouseDownInBackdrop mouseButtonNumber

Example Code

```
on mouseDownInBackdrop -- show a popup menu with a list of windows
  popup button "Windows"
end mouseDownInBackdrop
```

Comments

Handle the mouseDownInBackdrop message to perform an action when the user presses the mouse button while the mouse pointer is outside any window.

Parameters:

The mouseButtonNumber specifies which mouse button was pressed:

- 1 is the mouse button on Mac OS systems and the left button on Windows and Unix systems.
- 2 is the middle button on Unix systems.
- 3 is the right button on Windows and Unix systems and Control-click on Mac OS systems.

Comments:

The mouseDownInBackdrop message is sent to the current card.

If the backdrop property is set to "none", the mouseDownInBackdrop message is not sent.

mouseEnter

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

dragEnter message, focus command, mouseControl function, mouseLeave message, mouseLoc function, mouseMove message, mouseWithin message

Summary

Sent when the mouse pointer moves into an object.

Syntax

mouseEnter

Example Code

```
on mouseEnter -- show a Help field for the control the mouse is over
  show field (the short name of the target && "Help")
end mouseEnter
```

Comments

Handle the mouseEnter message to perform an action (for example, display additional information or highlight a button) when the mouse pointer enters an object.

Comments:

The mouseEnter message is sent only when the Browse tool is being used.

If two controls overlap, a mouseEnter message is sent whenever the mouse pointer crosses into a visible portion of a control. The control on the bottom receives a mouseEnter message only when the mouse pointer enters part of the control that can be seen. A control that is completely hidden by another control on top of it will never receive a mouseEnter message.

If the mouse button is down when the mouse pointer enters the control, no mouseEnter message is sent unless the mouse button is released while the pointer is still in the control.

If a control is shown (by changing its visible property to true or using the show command), and the mouse pointer is over the control when it is shown, Revolution sends a mouseEnter message to the control.

mouseH

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clickV function, mouseLoc function, mouseV function

Summary

Returns the horizontal position of the mouse pointer.

Syntax

the mouseH
mouseH()

Example Code

```
the mouseH  
add the mouseH to totalWidth
```

Comments

Use the mouseH function to find out where the mouse pointer is.

Value:

The mouseH function returns an integer.

Comments:

The returned value is the horizontal distance in pixels from the left edge of the current stack to the location of the mouse pointer. (Use the defaultStack property to identify the current stack.)

If the mouseH is positive, the pointer is to the right of the leftmost edge of the stack window. If the number is negative, the pointer is to the left of the stack window.

This function is equal to item 1 of the mouseLoc function.

Tip: The mouseMove message sends the mouse's horizontal position as a parameter. This means that in a mouseMove handler, normally it's not necessary to use the mouseH function.

mouseLeave

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

focus command, mouseControl function, mouseEnter message, mouseLoc function, mouseMove message, mouseWithin message

Summary

Sent when the mouse pointer moves out of an object.

Syntax

mouseLeave

Example Code

```
on mouseLeave -- hide another object
    hide field "Comments"
end mouseLeave
```

Comments

Handle the mouseLeave message to update a control when the mouse pointer moves outside it.

Comments:

The mouseLeave message is sent only when the Browse tool is being used.

If two controls overlap, a mouseLeave message is sent whenever the mouse pointer crosses outside a visible portion of a control. The control on the bottom receives a mouseLeave message only when the mouse pointer leaves part of the control that can be seen. A control that is completely hidden by another control on top of it will never receive a mouseLeave message.

If the mouse button is down when the mouse pointer leaves the control, the mouseLeave message is not sent until the mouse button is released.

If a control is hidden (by changing its visible property to false or using the hide command), and the mouse pointer is over the control when it is hidden, Revolution sends a mouseLeave message to the control.

mouseLine

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickLine function, foundLine function, mouseChar function, mouseCharChunk function, mouseControl function, mouseLoc function, mouseMove message, mouseStack function, mouseText function, selectedLine function

Summary

Returns a chunk expression describing the location of the line of text under the mouse pointer.

Syntax

the mouseLine

mouseLine()

Example Code

```
the mouseLine  
set the backgroundColor of the mouseLine to "yellow"
```

Comments

Use the mouseLine function within a handler to determine which line the mouse is hovering over, in order to take some action based on the click.

Value:

The mouseLine function returns a chunk expression of the form line lineNumber of field fieldNumber.

Comments:

The mouseLine function only returns locations in a field. If the mouse pointer is not over a field, the mouseLine function returns empty.

To get a chunk expression describing the exact word or text group that the mouse is over, use the mouseChunk function.

mouseLoc

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

click command, drag command, grab command, mouseDown message, mouseH function, mouseUp message, mouseV function, screenMouseLoc property, How to find out where the mouse pointer is, How to find out where the user clicked

Summary

Returns the location of the mouse pointer.

Syntax

the mouseLoc
mouseLoc()

Example Code

```
the mouseLoc  
click at the mouseLoc
```

Comments

Use the mouseLoc function to find out where the mouse is.

Value:

The mouseLoc function returns two integers separated by a comma.

Comments:

The first item of the return value is the horizontal distance in pixels from the left edge of the current stack to the location of the mouse. The second item of the return value is the vertical distance from the top edge of the current stack to the location of the mouse. (Use the defaultStack property to identify the current stack.)

The first item of the mouseLoc is equal to the mouseH. The second item is equal to the mouseV.

Tip: The mouseMove message sends the current mouse position as a parameter. This means that in a mouseMove handler, normally it's not necessary to use the mouseLoc function.

mouseMove

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

drag command, mouseEnter message, mouseLeave message, mouseLoc function, mouseWithin message

Summary

Sent when the user moves the mouse.

Syntax

mouseMove newMouseH,newMouseV

Example Code

```
on mouseMove -- highlight grouped text
  if the mouseChunk is not empty and

    "link" is among the items of the textStyle of the mouseChunk
  then set the backgroundColor of the mouseChunk to "yellow"
end mouseMove
```

Comments

Handle the mouseMove message if you want to perform some action (such as updating a field) when the user moves the mouse, or if you want to keep continuous track of the mouse pointer's position.

Parameters:

The newMouseH is the horizontal distance in pixels from the left edge of the stack to the mouse pointer's current location.

The newMouseV is the vertical distance in pixels from the top edge of the stack to the mouse pointer's current location.

Comments:

The mouseMove message is sent to the control the mouse pointer is over, or to the card if no control is under the mouse pointer. The parameters sent with the mouseMove message describe the mouse pointer's current location.

If the mouse button is down, the mouseMove message continues to be sent to the object that was clicked, even if the mouse pointer moves outside that object. However, if the mouse button was down at the time the pointer entered the control, no mouseMove messages are sent until the mouse button is released while the pointer is still in the control.

mouseRelease

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

click command, clickLoc function, dragDrop message, dragEnd message, mouse function, mouseDoubleDown message, mouseDown message, mouseUp message

Summary

Sent when the user releases the mouse outside the control that was clicked.

Syntax

mouseRelease mouseButtonNumber

Example Code

```
on mouseRelease theButton
  if theButton is 1 then send "mouseUp" to the target
  else pass mouseRelease
end mouseRelease
```

Comments

Handle the mouseRelease message to perform an action when the user releases the mouse button after clicking and then dragging outside the control.

Parameters:

The mouseButtonNumber specifies which mouse button was pressed:

- 1 is the mouse button on Mac OS systems and the left button on Windows and Unix systems.
- 2 is the middle button on Unix systems.
- 3 is the right button on Windows and Unix systems and Control-click on Mac OS systems.

Comments:

The mouseRelease message is sent to the control that was originally clicked, or to the card if no control was under the mouse pointer.

The mouseRelease message is sent only when the Browse tool is being used.

If an unlocked field is clicked with mouse button 1 or 2, no mouseRelease message is sent.

If the control is a field with its `listBehavior` property set to `true`, the `mouseRelease` message is sent when the mouse button is released and the mouse is not over a text line in the field, even if the mouse is still over the field.

If the mouse is released while it's still within the control that was clicked, a `mouseUp` message is sent instead of `mouseRelease`.

mouseStack

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickStack function, defaultStack property, focus command, mouseChar function, mouseCharChunk function, mouseControl function, mouseLine function, mouseLoc function, mouseMove message, mouseText function

Summary

Returns the name of the window the mouse pointer is over.

Syntax

the mouseStack

mouseStack()

Example Code

```
the mouseStack  
if the mouseStack is "Message box" then hide message
```

Comments

Use the mouseStack function to determine which window the mouse is over, or, in conjunction with the mouseControl, mouseLine, or mouseChunk, to determine which field the mouse is over.

Value:

The mouseStack function returns the short name property of the window the mouse is over.

Comments:

If the mouse is not over any Revolution window, the mouseStack returns empty.

If the mouse is over a stack's title bar or border, the mouseStack returns empty.

mouseStillDown

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

click command, clickLoc function, mouse function, mouseDoubleClickDown message, mouseDown message, mouseRelease message, mouseUp message

Summary

Sent periodically while the mouse button is being held down.

Syntax

mouseStillDown mouseButtonNumber

Example Code

Comments

You can handle the mouseStillDown message to perform an action when the user holds the mouse button down.

Parameters:

The mouseButtonNumber specifies which mouse button was pressed:

- 1 is the mouse button on Mac OS systems and the left button on Windows and Unix systems.
- 2 is the middle button on Unix systems.
- 3 is the right button on Windows and Unix systems and Control-click on Mac OS systems.

Comments:

The mouseStillDown message is sent to the control that was originally clicked, or to the card if no control was under the mouse pointer.

The period between mouseStillDown messages is specified by the idleRate and idleTicks properties.

The mouseStillDown message is sent only when the Browse tool is being used. If an unlocked field is clicked with mouse button 1 or 2, no mouseStillDown message is sent.

Usually, it is easier and more efficient to use the `mouseMove` message to track the movement of the mouse while the button is being held down.

Note: If there is no `mouseStillDown` handler in the target object's script, no `mouseStillDown` message is sent, even if there is a `mouseStillDown` handler in an object that's further along the message path.

mouseText

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clickText function, foundText function, mouseChar function, mouseCharChunk function, mouseControl function, mouseLine function, mouseLoc function, mouseMove message, mouseStack function, selectedText function

Summary

Returns the word or text group under the mouse pointer.

Syntax

the mouseText
mouseText()

Example Code

```
the mouseText  
if the style of the mouseText is "group" then showDef the mouseText
```

Comments

Use the mouseText function within a handler to determine which word or text group the mouse pointer is hovering over, in order to take some action based on the click.

Value:

The mouseText function returns the text that the mouse is over.

Comments:

The mouseText function only returns words in a field. If the mouse pointer is not over a field, the mouseText function returns empty.

The return value contains the word the mouse pointer is over. If the textStyle of the text is "link", the return value contains the entire text group.

To get the location of the word or text group clicked, use the mouseChunk function.

Important! Words are defined a little differently by the `mouseText` function than the way they are used in chunk expressions. A word, for purposes of the `mouseText`, is any text delimited by spaces, tabs, returns, or punctuation. If the mouse pointer is over a punctuation character, only that character is returned.

mouseUp

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

click command, clickLoc function, dragDrop message, dragEnd message, mouseDoubleUp message, mouseDown message, mouseRelease message, How to find out which object the user clicked, How to simulate a mouse click, How to throw away unwanted mouse clicks, Why does a mouseUp handler fail to work?

Summary

Sent when the user releases the mouse button.

Syntax

mouseUp mouseButtonNumber

Example Code

```
on mouseUp
    answer "You clicked" && the name of the target
end mouseUp
```

Comments

Handle the mouseUp message to perform an action when the user releases the mouse button after clicking.

Parameters:

The mouseButtonNumber specifies which mouse button was pressed:

1 is the mouse button on Mac OS systems and the left button on Windows and Unix systems.

2 is the middle button on Unix systems.

3 is the right button on Windows and Unix systems and Control-click on Mac OS systems.

Comments:

The mouseUp message is sent to the control that was clicked, or to the card if no control was under the mouse pointer.

The mouseUp message is sent only when the Browse tool is being used. If an unlocked field is clicked with mouse button 1 or 2, no mouseUp message is sent.

If the mouse has moved outside the control that was originally clicked before the user releases the mouse button, the mouseRelease message is sent instead of mouseUp. If the click is the second click of a double click, the mouseDoubleUp message is sent instead of mouseUp.

Tip: If the user clicks several times in rapid succession (for example, if the user clicks an "Increase" button that increases a number by 1), some of the clicks may send a mouseDoubleUp message instead of mouseUp. If your script only handles the mouseUp message, these accidental double clicks will be lost. One way to prevent this is to install a handler in an object that's further in the message path, to re-send double clicks:

```
on mouseDoubleUp
  if "on mouseUp" is in the script of the target

    and "on mouseDoubleUp" is not in the script of the target
    then send "mouseUp: to the target
  end mouseDoubleUp
```

If the user double-clicks an object whose script contains a mouseUp handler but no mouseDoubleUp, the above handler will automatically send a mouseUp to the object so the second click can be handled normally (instead of as a double-click).

Important! If the user clicks a transparent pixel in an image, the mouseUp message is sent to the object behind the image, not to the image.

mouseUpInBackdrop

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backdrop property, mouseDownInBackdrop message, mouseUp message

Summary

Sent when the user releases the mouse button. while the mouse pointer is in the backdrop.

Syntax

mouseUpInBackdrop mouseButtonNumber

Example Code

```
on mouseUpInBackdrop
  go to stack "Main" -- brings to front, if already open
end mouseUpInBackdrop
```

Comments

Handle the mouseUpInBackdrop message to perform an action when the user releases the mouse button after clicking in the backdrop.

Parameters:

The mouseButtonNumber specifies which mouse button was pressed:

• 1 is the mouse button on Mac OS systems and the left button on Windows and Unix systems.

• 2 is the middle button on Unix systems.

• 3 is the right button on Windows and Unix systems and Control-click on Mac OS systems.

Comments:

The mouseUpInBackdrop message is sent to the current card.

If the backdrop property is set to "none", the mouseUpInBackdrop message is not sent.

mouseV function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clickV function, mouseH function, mouseLoc function, mouseStack function, screenMouseLoc property

Summary

The mouseV property returns the vertical position of the pointer.

Syntax

the mouseV
mouseV()

Example Code

```
the mouseV  
put the mouseV into savedLocation
```

Comments

Use the mouseV function to find out where the mouse pointer is.

Value:

The mouseV function returns an integer.

Comments:

The returned value is the vertical distance in pixels from the top edge of the current stack to the location of the mouse pointer. (Use the defaultStack property to identify the current stack.)

If the mouseV is positive, the pointer is below the top of the stack window. If the number is negative, the pointer is above the stack window.

This function is equal to item 2 of the mouseLoc function.

Tip: The mouseMove message sends the mouse's vertical position as a parameter. This means that in a mouseMove handler, normally it's not necessary to use the mouseV function.

mouseWithin

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

focus command, mouseControl function, mouseEnter message, mouseLeave message, mouseLoc function, mouseMove message

Summary

Sent periodically to an object while the mouse pointer is within its borders.

Syntax

mouseWithin

Example Code

Comments

Handle the mouseWithin message to perform some repeating task (such as an update) for as long as the mouse pointer is over a control.

Comments:

The period between mouseWithin messages is specified by the idleRate and idleTicks properties.

The mouseWithin message is sent only when the Browse tool is being used.

If two controls overlap, a mouseWithin message is sent whenever the mouse pointer is in a visible portion of a control. The control on the bottom receives a mouseWithin message only when the mouse pointer is in a portion of the control that can be seen. A control that is completely hidden by another control on top of it will never receive a mouseWithin message.

If the mouse button is down when the mouse pointer enters the control, no mouseWithin messages are sent until the mouse button is released. If the mouse pointer leaves the control while the mouse button is still down, no mouseWithin messages are sent. If the mouse button is pressed while the pointer is in the control, however, mouseWithin messages continue to be sent, even while the mouse button is down.

Usually, it is easier and more efficient to use the `mousemove` message to track the movement of the mouse while the button is being held down. The `mouseover` message is sent continually and must be handled several times a second, taking up a great deal of processor time. The `mousemove` message is sent only when the mouse is moved, making it more efficient.

Note: If there is no `mouseover` handler in the target object's script, no `mouseover` message is sent, even if there is a `mouseover` handler in an object that's further along the message path.

move

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

drag command, location property, lockMoves property, moveSpeed property, moveStopped message, movingControls function, number property, stop moving command, syncRate property, How to animate a sprite, How to move cards from one stack to another

Summary

Moves an object from one location to another.

Syntax

move object {[from startLoc] to endLoc|to pointList|rel[ative] motion}

[in time] [without {messages | waiting}]

Example Code

```
move image "sprite" to the mouseLocation in 30 ticks
move button ID 4 relative 50,50 without waiting
move stack "Overview Palette" from 100,30 to 0,30 without messages
move the mouseControl to the points of graphic "Polygon" in 10 seconds
```

Comments

Use the move command to animate controls or windows by moving them smoothly across the screen.

Parameters:

The object is any visible control or open stack reference. (You can specify a card with the move command, but moving a card has no effect.)

The startLoc is an expression that evaluates to a point—a vertical and horizontal distance from the top left of the current stack, separated by a comma. (If the object is a stack, the distance is from the top left of the screen.) The object is shown at the startLoc before the move begins. If no startLoc is specified, the move starts at the object's current location.

The endLoc is an expression that evaluates to a point.

The pointList is a return-separated list of destination points. The object is moved to each point in turn during the move.

The motion consists of a horizontal number of pixels and a vertical number of pixels, separated by a comma. A positive number moves the object to the left or down; a negative number moves it to the right or up.

The time specifies a total time for the move to take from start to end, in milliseconds, seconds, or ticks. If you do not specify a time, the speed of the move is determined by the moveSpeed property.

Comments:

Use the move...relative form of the move command to move an object in a straight line a certain distance from its current location. Use the move...from startLoc to endLoc form to move an object in a straight line from one location to another, without regard to where the object started. Use the move...to pointList form to move an object along a set of defined points.

If you specify without messages, built-in messages are not delivered during the move.

If you specify without waiting, the current handler continues immediately, and a moveStopped message is sent when the move is completed. Otherwise, the handler pauses until the move is complete. If you issue another move command while a previous move command for the same object is still executing, the previous move command is halted, and the second move command starts up at the object's current location.

To move multiple objects at the same time, set the lockMoves property to true before issuing the move commands. When you set the lockMoves back to false, all the pending moves begin at once.

Tip: To easily move an object along a curved path, create a curve graphic using the freehand Curve tool, then use a statement like the following to move the object:

```
move button "My Button" to the points of graphic "My Curve"
```

If you like, you can hide the graphic so that the object follows the invisible curve, but its path is not visible on screen.

moveControl

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

grab command, location property, moveStack message, rectangle property, resizeControl message

Summary

Sent to a control that the user moved with the Pointer tool.

Syntax

moveControl

Example Code

```
on moveControl -- in a button script
  -- move the same-named field to be below the moved button:
  set the topLeft of field (the short name of the target)

    to the bottomLeft of the target
end moveControl
```

Comments

Handle the moveControl message if you want to respond to the user's movement of a control. For example, you can create a moveControl handler that responds to movement of a button by moving fields out of the button's way.

Comments:

The moveControl message is sent only if the user moved the control. This message is not sent when a handler moves the control by changing its location, rectangle, or other properties.

The moveControl message is sent after the movement is finished. This means that you cannot prevent a control from being moved by trapping this message.

moveSpeed

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

dragSpeed property, lockMoves property, move command

Summary

Specifies how fast the move command moves objects.

Syntax

set the moveSpeed to pixelsPerSecond

Example Code

```
set the moveSpeed to 20 -- slow
```

Comments

Use the moveSpeed property to change the speed of animation performed with the move command.

Value:

The moveSpeed is an integer between zero and 65535.

By default, the moveSpeed property is set to 200.

Comments:

A moveSpeed of zero moves the objects directly to their destination, as though you had set the object's location property rather than using the move command.

The moveSpeed property has no effect if you include the in time parameter in the move command. It only controls the speed of move commands that don't specify a time for the movement.

moveStack

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

grab command, location property, moveControl message, rectangle property, resizeControl message, How to prevent the user from moving a window, How to respond to moving a window

Summary

Sent to the current card when the user moves the stack window.

Syntax

moveStack newStackH,newStackV

Example Code

```
on moveStack finalLeft, finalTop -- move ancillary window
  set the right of stack "Palette" to finalLeft - 5
  set the top of stack "Palette" to finalTop
end moveStack
```

Comments

Handle the moveStack message if you want to respond to movement of the stack window.

Parameters:

The newStackH is the horizontal distance in pixels from the left of the screen to the left edge of the stack
.

The newStackV is the vertical distance in pixels from the top of the screen to the top edge of the stack.

Comments:

The moveStack message is sent only if the user moved the stack window. This message is not sent when a handler moves the window by changing its location, rectangle, or other properties.

The moveStack message is sent after the movement is finished. This means that you cannot prevent a stack window from being moved by trapping this message.

moveStopped

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

move command, movingControls function, stop moving command

Summary

Sent to an object that has just been moved by the move command.

Syntax

moveStopped

Example Code

```
on moveStopped -- play a sound
  play audioClip "Finished"
end moveStopped
```

Comments

Handle the moveStopped message to coordinate an action with the end of a move command.

Comments:

The moveStopped message is sent when the actions of the move command end, or when a movement is ended by the stop moving command.

movie function

Synonyms
movies

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

done constant, play command, sound function, templateVideoClip keyword

Summary

Returns the name of the currently playing video clip.

Syntax

the movie
movie()

Example Code

```
the movie  
repeat until myMovie is not among the lines of the movies
```

Comments

Use the movie function to synchronize other actions with sounds.

Value:

The movie function returns a list of video clip names, one per line.

Comments:

If no video clip is playing, the movie function returns done.

movieControllerID

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

play command, showController property, windowID property

Summary

Reports a pointer to a movie's controller.

Syntax

get the movieControllerID of player

Example Code

```
put the movieControllerID of player "Intro" into field "IDs"  
get myLink(the movieControllerID of player 3) -- an example external
```

Comments

Use the movieControllerID property to pass to an external that needs to manipulate the player.

Value:

The movieControllerID of a stack is an integer.

The movieControllerID property is read-only and cannot be set.

Comments:

The controller ID is provided by the operating system.

Some externals need to manipulate the QuickTime movie directly, and do so by referencing the ID. Such externals require you to pass the ID when you use the external.

movingControls

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

lockMoves property, move command, moveStopped message, stop moving command

Summary

Returns a list of controls that are being moved by the move command.

Syntax

the movingControls
movingControls()

Example Code

```
the movingControls  
if myID is among the lines of the movingControls then beep
```

Comments

Use the movingControls function to find out whether a control or window is still moving, or to perform an action on each control or window that is moving.

Value:

The movingControls function returns a list of the long ID properties of each moving control or window, one per line.

multiEffect

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

visual effect command

Summary

Has no effect and is included in Transcript for compatibility with imported SuperCard projects.

Syntax

set the multiEffect to {true | false}

Example Code

Comments

In SuperTalk, the multiEffect property determines whether multiple visual effects are played in succession or interleaved. In Revolution, visual effects are always played back in succession, one after the other.

The multiEffect property is always set to false. A handler can set it to any value without causing a script error, but the actual value is not changed.

multiple

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

centered property, lineSize property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the multiple to {true | false}

Example Code

Comments

In HyperCard, the multiple property determines whether multiple objects are drawn when using the paint tools. In Revolution, the paint tools always draw a single shape.

multipleLines

property

Synonyms

multipleHilites

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hilitedLine property, listBehavior property, noncontiguousHilites property, selectedText function, toggleHilites property

Summary

Specifies whether more than one line can be selected in a list field.

Syntax

set the multipleLines of field to {true | false}

Example Code

```
set the multipleLines of last card field to true
```

Comments

Use the multipleLines property to control the behavior of list fields.

Value:

The multipleLines of a field is true or false.

By default, the multipleLines property of newly created fields is set to false.

Comments:

If the multipleLines of a field is set to true, the user can select more than one line by dragging within the field, or Shift-clicking to select all the lines between the selected line and the clicked line. If the multipleLines is false, clicking a line deselects any other lines in the field.

If the field's listBehavior property is false, the multipleLines property has no effect.

Important! Setting the toggleHilites to true automatically sets the field's multipleLines property to true. (However, setting the toggleHilites to false does not set the property back to false.) To set the toggleHilites to true and the multipleLines to false, be sure to set the toggleHilites first, then set the multipleLines to false.

multiply command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

* operator, add command, divide command, matrixMultiply function, numberFormat property, subtract command

Summary

Multiplies a number by a container and places the resulting value in the container.

Syntax

multiply [chunk of] container by number

multiply arrayContainer by {number | array}

Example Code

```
multiply field 17 by it
multiply yearlyTotals by yearlyPercentages
multiply line thisLine of myVariable by theFactor
```

Comments

Use the multiply command to multiply a container or a portion of a container by a number, or to multiply two arrays containing numbers.

Parameters:

The chunk is a chunk expression specifying a portion of the container.

The container is a field, button, or variable, or the message box.

The number is any expression that evaluates to a number.

The arrayContainer is an array variable each of whose elements is a number.

Comments:

The contents of the container (or the chunk of the container) must be a number or an expression that evaluates to a number.

If an arrayContainer is multiplied by a number, each element is multiplied by the number. If an arrayContainer is multiplied by an array, both arrays must have the same number of elements and the same dimension, and each element in the arrayContainer is multiplied by the corresponding element of the array.

If the container or an element of the arrayContainer is empty, the multiply command treats its contents as zero.

If container is a field or button, the format of the result is determined by the numberFormat property.

Changes to Transcript:

The multiply arrayContainer form was introduced in version 1.1. In previous versions, only single numbers could be used with the multiply command.

multiSpace

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

multiple property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the multiSpace to {true | false}

Example Code

Comments

In HyperCard, the multiSpace property determines the amount of space left between multiple paint shapes. In Revolution, the paint tools draw a single shape, so the multiSpace property has no effect.

name
property

Synonyms

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
cardNames property, fileName property, groupNames property, ID property, label property, nameChanged message, number property, showName property, About object types and object references, How to refer to a control on another card, How to rename a custom property, Why can't I find a stack I just saved?, Why is the card number in the window name?, Why is there an asterisk in the window name?

Summary
Specifies the name of an object.

Syntax
the [long | abbr[ev[iated]] | short] name of object

Example Code
set the name of the target to it
set the name of last card button to "Bing!"

Comments
Use an object's name property to refer to the object.

Value:
The name of an object is a string.

Comments:
The short name of an object is simply the name that the object has been assigned. When you set an object's name property or type it into the object's property inspector, the short name is what you provide.

The abbreviated name of an object is the object's type followed by the object's short name in quotes. For example, if a player's short name is Heaven, its abbreviated name is player "Heaven". If you don't specify a modifier for the name property, the abbreviated name is what you get.

The long name of an object includes information about its owner (and about the owner of that object, and so forth). For example, suppose a stack named "My Stack" contains a card named "My Card". This

card has a group named "My Group" which contains a button named "My Button". The card also has a card field named "My Field". If "My Stack" is a main stack and it's in a file whose path is "/Drive/Folder/Stack.rev", the long names of these objects look like this:

ï The stack:

stack "/Drive/Folder/Stack.rev"

ï The group:

group "My Group" of stack "/Drive/Folder/Stack.rev"

ï The card:

card "My Card" of stack "/Drive/Folder/Stack.rev"

ï The grouped button:

button "My Button" of group "My Group" of card "My Card" of stack "/Drive/Folder/Stack.rev"

ï The card field:

field "My Field" of card "My Card" of stack "/Drive/Folder/Stack.rev"

Important! On OS X systems, standalone applications are stored as application bundles. A bundle behaves like a file but is actually a folder, and the main stack of a standalone application is inside this folder. The long name of a stack in an application is the location of the application inside the bundle, not the bundle's location. For example, if the bundle's file path is "/Volumes/Disk/MyApp.app/", the long name of the application's main stack might be "/Volumes/Disk/MyApp.app/Contents/MacOS/MyApp".

If the stack is a substack, its name is included in the long names of its objects, before the path of the main stack:

stack "My Substack" of stack "/Drive/Folder/Stack.rev"

The long name of a group includes the name of the current card. If the group does not appear on the current card, requesting its name results in an execution error. If you need to get the name of a group that is not on the current card, use the "background" terminology instead.

The long name of a background includes the name of the current card, if the background appears on the current card. If not, the long name of the background includes the name of the first card the background appears on.

You can use an object's name to refer to the object in a statement. However, the name need not be seen on screen, even if the object is one that normally has a visible name (such as a button). If a stack or control's label property is not empty, the label property is used when a visible label is displayed. If a stack's label property is empty, its name is used as the window title. Likewise, if a control's label property is empty, its name is used as the label. This means that you can use any string you want for the name without worrying about the user-interface impact on your application: if the name is not user-friendly, set the label to something that is.

A stack's name cannot be empty. (Other objects can have empty names.) Attempting to set a stack's name to empty does not rename it.

If an object's name is empty, getting its name yields its ID property instead. In this case, the abbreviated ID form is always reported, regardless of what form of the name property you request.

Caution! Avoid naming an object with a number. Doing so may cause Revolution to become confused by object references that specify that number, since they might be referring to the object by name or to a different object by number.

Caution! Avoid giving two controls in the same stack the same name. Doing so may cause problems if you use the Geometry pane in the property inspector or the Animation Builder.

Caution! Avoid using "rev" as the first three characters of a stack's name. These characters are reserved for stacks that are part of the Revolution development environment, and giving a non-Revolution stack such a name may cause menus or buttons to be inappropriately disabled, or other problems. You can use the characters "rev" anywhere in a stack name except the beginning. If you want a name beginning with "rev" to appear in a stack's title bar, set the stack's label property instead.

Note: If a stack's `hcAddressing` property is set to true, the long or abbreviated name of a control in that stack begins with the word "background" if the control is part of a group, and with the word "card" if not.

nameChanged

message

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cardNames property, groupNames property, IDChanged message, name property, number property, showName property

Summary

Sent to an object when its name is changed.

Syntax

nameChanged oldName,newName

Example Code

```
on nameChanged myOldName,myNewName -- replace name in a menu button
  put lineOffset(myOldName,button "Window List") into namePosition
  put myNewName into line namePosition of button "Window List"
end nameChanged
```

Comments

Handle the nameChanged message if you want to make updates when an object's name is changed. For example, if you have a popup menu with the names of stacks, the menu should be updated when a stack's name changes.

Parameters:

The oldName is the object's original name.

The newName is the object's new name.

Comments:

The actual change is not triggered by the nameChanged message, so trapping the message and not allowing it to pass does not prevent the name from being changed.

navigationArrows

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

arrowKey message, go command, textArrows property, How to block or change the action of arrow keys, Edit menu > Preferences

Summary

Specifies whether the arrow keys move from card to card.

Syntax

set the navigationArrows to {true | false}

Example Code

```
set the navigationArrows to true
```

Comments

Use the navigationArrows property to control what the arrow keys do.

Value:

The navigationArrows is true or false.

By default, the navigationArrows property is set to true.

Comments:

If the navigationArrows is true, pressing the arrow keys moves to a different card:

- The Left arrow key moves to the next card.
- The Right arrow key moves to the previous card.
- The Up arrow key moves backward within the recent-cards list.
- The Down arrow key moves forward within the recent-cards list.

If there is an insertion point or text selection in a field, and the textArrows property is set to true, the arrow keys move the insertion point one character left or right, or one line up or down. In this case, the setting of the navigationArrows property has no effect.

If an object is selected, pressing the arrow keys nudges the object one pixel in the appropriate direction.

newBackground

message

Synonyms

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clone command, copy command, create command, deleteBackground message, group command, newCard message, newGroup message, newStack message, place command, templateGroup keyword

Summary

Sent to a new group right after it's created.

Syntax

newBackground

Example Code

Comments

Handle the newBackground message if you want to create objects or do other tasks when a new group is created.

Comments:

Normally, the newBackground message is handled at a higher level of the message path, since the newly created group does not yet have a script unless the templateGroup contains one.

The actual creation is not triggered by the newBackground message, so trapping the message and not allowing it to pass does not prevent the group from being created.

A newGroup message is sent before the newBackground message. (The newBackground message is included for compatibility with HyperCard.)

newButton

message

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clone command, copy command, create command, deleteButton message, templateButton keyword

Summary

Sent to a new button right after it's created.

Syntax

newButton

Example Code

```
on newButton -- put new buttons at a random location
  set the location of the target to

  random(the width of this stack - the width of the target div 2),

  random(the height of this stack - the height of the target div 2)
end newButton
```

Comments

Handle the newButton message if you want to create additional objects, switch tools, or do other tasks when a new button is created.

Comments:

Normally, the newButton message is handled at a higher level of the message path, since the newly created button does not yet have a script unless the templateButton contains one.

The actual creation is not triggered by the newButton message, so trapping the message and not allowing it to pass does not prevent the button from being created.

newCard

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clone command, copy command, create command, deleteCard message, newBackground message, newGroup message, newStack message, place command, templateCard keyword

Summary

Sent to a new card right after it's created.

Syntax

newCard

Example Code

```
on newCard -- ask the user to name the new card
  ask "What do you want to call this card?"
  if the result is "Cancel" then delete the target
  else set the name of the target to it
end newCard
```

Comments

Handle the newCard message if you want to create additional objects, switch tools, or do other tasks when a new card is created.

Comments:

Normally, the newCard message is handled at a higher level of the message path, since the newly-created card does not yet have a script unless the templateCard contains one.

The actual creation is not triggered by the newCard message, so trapping the message and not allowing it to pass does not prevent the card from being created.

newEPS

message

Synonyms

Objects

EPS

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clone command, copy command, create command, deleteEPS message, templateEPS keyword

Summary

Sent to a new EPS object right after it's created.

Syntax

newEPS

Example Code

```
on newEPS -- warn the user if not supported
  if the platform is "MacOS" or the platform is "Win32" then
    answer "EPS objects are not supported on this system." with "Sorry"
  end if
end newEPS
```

Comments

Handle the newEPS message if you want to create additional objects, switch tools, or do other tasks when a new EPS object is created.

Comments:

Normally, the newEPS message is handled at a higher level of the message path, since the newly-created EPS object does not yet have a script unless the templateEPS contains one.

The actual creation is not triggered by the newEPS message, so trapping the message and not allowing it to pass does not prevent the EPS object from being created.

newField

message

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clone command, copy command, create command, deleteField message, templateField keyword

Summary

Sent to a new field right after it's created.

Syntax

newField

Example Code

```
on newField -- create a label field
  ask "What do you want to label this field?"
  if it is not empty then
    put it into myLabel
    set the name of the target to myLabel
    lock messages
    create field
    put myLabel into last field
  end if
end newField
```

Comments

Handle the newField message if you want to create additional objects, switch tools, or do other tasks when a new field is created.

Comments:

Normally, the newField message is handled at a higher level of the message path, since the newly-created field does not yet have a script unless the templateField contains one.

The actual creation is not triggered by the newField message, so trapping the message and not allowing it to pass does not prevent the field from being created.

newGraphic

message

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clone command, copy command, create command, deleteGraphic message, templateGraphic keyword

Summary

Sent to a new graphic right after it's created.

Syntax

newGraphic

Example Code

```
on newGraphic
  if the style of the target is "line" then set the points
    of the target to the storedCoordinates of this card
end newGraphic
```

Comments

Handle the newGraphic message if you want to create additional objects, switch tools, or do other tasks when a new graphic is created.

Comments:

Normally, the newGraphic message is handled at a higher level of the message path, since the newly created graphic does not yet have a script unless the templateGraphic contains one.

The actual creation is not triggered by the newGraphic message, so trapping the message and not allowing it to pass does not prevent the graphic from being created.

newGroup

message

Synonyms

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clone command, copy command, create command, deleteBackground message, group command, newBackground message, newCard message, newStack message, place command, templateGroup keyword

Summary

Sent to a new group right after it's created.

Syntax

newGroup

Example Code

```
on newGroup -- automatically set a property of all new groups
    set the backgroundColor of the target to true
end newGroup
```

Comments

Handle the newGroup message if you want to create objects or do other tasks when a new group is created.

Comments:

Normally, the newGroup message is handled at a higher level of the message path, since the newly-created group does not yet have a script unless the templateGroup contains one.

The actual creation is not triggered by the newGroup message, so trapping the message and not allowing it to pass does not prevent the group from being created.

A newBackground message is sent after the newGroup message. (The newBackground message is included for compatibility with HyperCard.)

newImage

message

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clone command, copy command, create command, deleteImage message, templateImage keyword

Summary

Sent to a new image right after it's created.

Syntax

newImage

Example Code

```
on newImage -- assign a file to a referenced image
  if the text of the target is empty then -- no painted pixels
    answer file "Please choose an image to display:"
    set the filename of the target to it
  end if
end newImage
```

Comments

Handle the newImage message if you want to create additional objects, switch tools, or do other tasks when a new image is created.

Comments:

Normally, the newImage message is handled at a higher level of the message path, since the newly-created image does not yet have a script unless the templateImage contains one.

The actual creation is not triggered by the newImage message, so trapping the message and not allowing it to pass does not prevent the image from being created.

newPlayer

message

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clone command, copy command, create command, deletePlayer message, templatePlayer keyword

Summary

Sent to a new player right after it's created.

Syntax

newPlayer

Example Code

```
on newPlayer -- assign a file
  if the filename of the target is empty then
    answer file "Please choose a file to play:"
    set the filename of the target to it
  end if
end newImage
```

Comments

Handle the newPlayer message if you want to create additional objects, switch tools, or do other tasks when a new player is created.

Comments:

Normally, the newPlayer message is handled at a higher level of the message path, since the newly-created player does not yet have a script unless the templatePlayer contains one.

The newly-created player's long ID property is placed in the it variable.

The actual creation is not triggered by the newPlayer message, so trapping the message and not allowing it to pass does not prevent the player from being created.

newScrollbar

message

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clone command, copy command, create command, deleteScrollbar message, templateScrollbar keyword

Summary

Sent to a new scrollbar right after it's created.

Syntax

newScrollbar

Example Code

```
on newScrollbar -- store its ID in a custom property of the card
  set the storedID of this card to the ID of the target
end newScrollbar
```

Comments

Handle the newScrollbar message if you want to create additional objects, switch tools, or do other tasks when a new scrollbar is created.

Comments:

Normally, the newScrollbar message is handled at a higher level of the message path, since the newly-created scrollbar does not yet have a script unless the templateScrollbar contains one.

The actual creation is not triggered by the newScrollbar message, so trapping the message and not allowing it to pass does not prevent the scrollbar from being created.

newStack

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeStack message, create command, deleteStack message, newCard message, templateStack keyword

Summary

Sent to the current card of a new stack right after the stack is created.

Syntax

newStack

Example Code

```
on newStack -- store a path for later use in a closeStack handler
  ask file "Where do you want to store the backup?" with

    char 1 to 24 of the short name of this stack && "Backup"
  set the backupLocation of the target to it
end newStack
```

Comments

Handle the newStack message if you want to create additional objects or do other tasks when a new stack is created.

Comments:

Normally, the newStack message is handled at a higher level of the message path, since the newly-created stack does not yet have a script unless the templateStack contains one.

The actual creation is not triggered by the newStack message, so trapping the message and not allowing it to pass does not prevent the stack from being created.

newTool

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

choose command, tool property

Summary

Sent to the current card when a different tool is chosen.

Syntax

newTool toolName

Example Code

```
on newTool newTool
  if newTool is "Browse" then show stack "Editing Tools"
  else hide stack "Editing Tools"
end newTool
```

Comments

Handle the newTool message if you want to change something when a tool is chosen. For example, you might want to show a particular palette window only with the Browse tool.

Parameters:

The toolName is the name of the tool that has just been chosen.

Comments:

The newTool message is sent when the user chooses a tool from the Tools palette or when a handler chooses a new tool with the choose command.

next

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

first keyword, last keyword, next repeat control structure, previous keyword

Summary

Designates the card following the current card.

Syntax

Example Code

```
go next marked card
```

Comments

Use the next keyword to move forward in the stack or to reference the card after the current card.

Comments:

If you are already on the last card, the next card is the first card in the stack.

next repeat

control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

end repeat keyword, exit repeat control structure, next keyword, repeat control structure

Summary

Skips the rest of the current iteration and goes back to the top of the repeat structure.

Syntax

next repeat

Example Code

Comments

Use the next repeat control structure to skip part of a repeat loop.

Form:

The next repeat statement appears on a line by itself, anywhere inside a repeat control structure.

Comments:

The next repeat control structure skips the rest of the current iteration, and continues to the next iteration. The following example performs the loop action only if the current card's name contains the letter "e":

```
repeat for the number of cards of this stack
  go to next card
  if "e" is in the short name of this card then next repeat
  put the short name of this card & return after myCardsList
end repeat
```

Usually, next repeat is used within an if control structure, so that the current iteration is skipped if a condition is true and continues if the condition is false.

nine

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command, iBeam constant, ninth keyword

Summary

Equivalent to the number 9.

Syntax

Example Code

```
if offset(it,pathName) > nine then put true into longFldr
```

Comments

Use the nine constant when it is easier to read than the numeral 9.

ninth

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

last keyword, middle keyword, ninth keyword, number property

Summary

Designates the ninth member of a set.

Syntax

Example Code

```
set the script of the ninth group to empty  
get first item of ninth line of field "Expressions"
```

Comments

Use the ninth keyword in an object reference or chunk expression.

Comments:

The ninth keyword can be used to specify any object whose number property is 9. It can also be used to designate the ninth chunk in a chunk expression.

The word the is optional when using the ninth keyword.

nodeChanged

message

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

currentNode property, hotspotClicked message, nodes property, QTDebugStr message

Summary

Sent to a player containing a QuickTime VR movie when the current node is changed.

Syntax

nodeChanged newNodeID

Example Code

```
on nodeChanged theNode -- show some text from a custom property
  put the nodeDescription[theNode] of me into field "Description"
end nodeChanged
```

Comments

Handle the nodeChanged message if you want to make updates when the user navigates to a new node in a QuickTime VR movie.

Parameters:

The newNodeID is the ID of the QuickTime VR node being moved to.

Comments:

If the user navigates to another node by clicking a hot spot in the QuickTime VR movie, the hotspotClicked message is sent before the nodeChanged message.

A handler can change the node by setting the player's currentNode property.

nodes

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

currentNode property, hotspots property, nodeChanged message, play command, trackCount property

Summary

Reports a list of the nodes in a QuickTime VR movie.

Syntax

get the nodes of player

Example Code

```
repeat for each line in the nodes of player "Arctic"
if item 2 of line 1 of the nodes of me is "panorama" then zoomOut
```

Comments

Use the nodes property to list the nodes of a QuickTime VR movie.

Value:

The nodes consists of a list of nodes, one per line. Each line consists of two items, separated by a comma:

- the node ID (an integer)

- the node type (either "object" or "panorama")

The nodes property is read-only and cannot be set.

Comments:

Each node of a QuickTime VR movie is a viewpoint. The movie author sets the nodes during development of the movie. The user can change nodes using the navigational controls in the player; a handler can change nodes by setting the player's currentNode property.

If the player does not contain a QuickTime VR movie, its nodes property is empty.

noncontiguousHilites

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

hilitedLine property, listBehavior property, multipleLines property, toggleHilites property

Summary

Specifies whether the user can select non-adjacent lines of a list field.

Syntax

set the noncontiguousHilites of field to {true | false}

Example Code

```
set the noncontiguousHilites of field "Parties" to true
```

Comments

Use the noncontiguousHilites property to control the behavior of list fields.

Value:

The noncontiguousHilites of a field is true or false.

By default, the noncontiguousHilites property of newly-created fields is set to false.

Comments:

If a list field's noncontiguousHilites property is false, the user can select multiple lines only if they're next to each other.

If the noncontiguousHilites is true, the user can select multiple lines, wherever they occur in the field. The user makes a multiple choice from a list field by Command-clicking the desired lines (on Mac OS systems) or Control-clicking (on Unix and Windows systems).

If the field's listBehavior is not set to true, this property has no effect.

Important! Setting the toggleHilites to true automatically sets the field's noncontiguousHilites property to true. (However, setting the toggleHilites to false does not set the property back to false.) To set the

toggleHilites to true and the noncontiguousHilites to false, be sure to set the toggleHilites first, then set the noncontiguousHilites to false.

noOp

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword, Shortcut to make an image transparent

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of field "Alerts" to noOp
```

Comments

Use the noOp keyword to make an object transparent.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

When the noOp transfer mode is used, the object's background and border are not drawn and seem to disappear, letting objects behind it show through. (Text is still drawn.)

Tip: Graphics whose ink is set to noOp are transparent, but (unlike graphics whose opaque property is set to false) they still intercept mouse clicks within the graphic's outline. To create an invisible hot spot with a non-rectangular shape, use a curve or polygon graphic and set its ink property to noOp.

normal

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

find command, visual effect command, words keyword, chars keyword, string keyword

Summary

Used with the visual effect command to designate a straight cut between two cards with no visual effect. Also used with the find command to search at the beginnings of one or more words.

Syntax

Example Code

```
find normal "run" -- finds "run" and "running", but not "grunt"  
visual effect normal to black -- cut to black
```

Comments

Use the normal keyword to improve the clarity of your code, or to provide a cut to one of the supported visual effect colors.

Comments:

Since normal is the default mode for the find command, you never need to actually use the normal keyword; if you leave it out, the search is performed in normal mode anyway.

When used with the find command, the normal keyword finds cards that contain each of the specified word beginnings, though not necessarily next to each other.

not
operator

Synonyms

Objects
logical

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
and operator, bitNot operator, or operator, Operator Precedence Reference

Summary
Negates a logical value.

Syntax
not value

Example Code
`not true -- evaluates to false`
`set the visible of me to not the visible of me`

Comments
Use the not operator to reverse the meaning of a logical expression.

Parameters:
The value is true or false, or an expression that evaluate to true or false.

Comments:
If the value is true, not value is false, and if the value is false, not value is true.

You can combine the logical operators and, or, and not in an expression.

notSrcAnd

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of card button 1 to notSrcAnd
```

Comments

Use the notSrcAnd keyword to invert the colors of an object and make the dark parts transparent.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color (also called the source color) is displayed.

The notSrcAnd transfer mode performs the following steps to compute the color of each pixel:

1. Each component of the object's coloróthe number indicating the amount of red, green, or blueóis changed to its inverse. (If the color is expressed as three integers between zero and 255óone for each of red, green, and blueóthen the inverse of each component is equal to 255 minus the component's value.)
2. Revolution performs a bitAnd operation on each component of the inverted object color with the corresponding component of the color under the object.

Each component of the resulting color is equal to the result of this expression:

(255 - object component) bitAnd background component

The effect is that the color of the object is reversed, and the darker an object is, the more transparent it is. Black parts of an object are completely transparent, and white parts are completely opaque.

For example, suppose an object's color is 120,0,255, and the color of the pixels under the object is 50,20,240. If the notSrcAnd transfer mode is used, the object's displayed color is 2,20,0.

The notSrcAnd mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear all-white.

notSrcAndReverse

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of card button 1 to notSrcAnd
```

Comments

Use the notSrcAndReverse keyword to invert the colors under an object and make the light parts opaque.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color (also called the source color) is displayed.

The notSrcAndReverse transfer mode performs the following steps to compute the color of each pixel:

1. Each component of the object's coloróthe number indicating the amount of red, green, or blueóis changed to its inverse. (If the color is expressed as three integers between zero and 255óone for each of red, green, and blueóthen the inverse of each component is equal to 255 minus the component's value.)
2. Each component of the color underneath the object is changed to its inverse.

3. Revolution performs a bitAnd operation on each component of the inverted object color with the corresponding component of the inverted color under the object.

Each component of the resulting color is equal to the result of this expression:
 $(255 - \text{object component}) \text{ bitAnd } (255 - \text{background component})$

The effect is that the color of the object is reversed, and the darker an object is, the more transparent it is. Black parts of an object are completely transparent, and white parts are completely opaque.

For example, suppose an object's color is 120,0,255, and the color of the pixels under the object is 50,20,240. If the notSrcAndReverse transfer mode is used, the object's displayed color is 135,235,0.

The notSrcAndReverse mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear as though their ink were set to srcCopy.

notSrcBic

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of image 7 to notSrcBic
```

Comments

Use the notSrcBic keyword to change the appearance of an object.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed. When the notSrcAnd mode is used, each component of the color underneath the object (red, green, and blue) is changed to be equal to the color underneath it. The object disappears except for its text.

The notSrcBic mode can be used only on Mac OS systems. On Unix and Windows systems, objects whose ink property is set to this mode appears as though their ink were set to srcCopy.

notSrcCopy

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of last image to notSrcCopy
```

Comments

Use the notSrcCopy keyword to turn an object white.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed. When the notSrcCopy mode is used, each component of the color underneath the object (red, green, and blue) is raised to be equal to white.

notSrcOr

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of button ID 9 to notSrcOr
```

Comments

Use the notSrcOr keyword to invert the colors of an object and make the dark parts transparent.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

The notSrcOr transfer mode performs the following steps to compute the color of each pixel:

1. Each component of the object's coloróthe number indicating the amount of red, green, or blueóis changed to its inverse. (If the color is expressed as three integers between zero and 255óone for each of red, green, and blueóthen the inverse of each component is equal to 255 minus the component's value.)
2. Revolution performs a bitOr operation on each component of the inverted object color with the corresponding component of the color under the object.

Each component of the resulting color is equal to the result of this expression:

(255 - object component) bitOr background component

The effect is that the color of the object is reversed, and the lighter an object is, the more transparent it is. White parts of an object are completely transparent, and black parts are completely opaque.

For example, suppose an object's color is 120,0,255, and the color of the pixels under the object is 50,20,240. If the notSrcOr transfer mode is used, the object's displayed color is 183,255,240.

The notSrcOr mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear as though their ink were set to noOp.

notSrcOrReverse

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of graphic "Flash" to notSrcOrReverse
```

Comments

Use the notSrcOrReverse keyword to invert the colors under an object and make the dark parts opaque.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

The notSrcAndReverse transfer mode performs the following steps to compute the color of each pixel:

1. Each component of the object's coloróthe number indicating the amount of red, green, or blueóis changed to its inverse. (If the color is expressed as three integers between zero and 255óone for each of red, green, and blueóthen the inverse of each component is equal to 255 minus the component's value.)
2. Each component of the color underneath the object is changed to its inverse.
3. Revolution performs a bitOr operation on each component of the inverted object color with the corresponding component of the inverted color under the object.

Each component of the resulting color is equal to the result of this expression:
 $(255 - \text{object component}) \text{ bitOr } (255 - \text{background component})$

The effect is that both the color of the object and the color under the object are reversed, and the lighter an object is, the more opaque it is. White parts of an object are completely opaque, and black parts are completely inverted.

For example, suppose an object's color is 120,0,255, and the color of the pixels under the object is 50,20,240. If the notSrcOrReverse transfer mode is used, the object's displayed color is 207,255,15.

The notSrcOrReverse mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear as though their ink were set to srcCopy.

notSrcXOr

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of scrollbar "Progress" to notSrcXOr
```

Comments

Use the notSrcXOr keyword to invert the colors of an object and make the light parts transparent.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

The notSrcXOr transfer mode performs the following steps to compute the color of each pixel:

1. Each component of the object's coloróthe number indicating the amount of red, green, or blueóis changed to its inverse. (If the color is expressed as three integers between zero and 255óone for each of red, green, and blueóthen the inverse of each component is equal to 255 minus the component's value.)
2. Revolution performs a bitXOr operation on each component of the inverted object color with the corresponding component of the color under the object.

Each component of the resulting color is equal to the result of this expression:

(255 - object component) bitXOr background component

The effect is that the color of the object is reversed, and the darker an object is, the more transparent it is.

For example, suppose an object's color is 120,0,255, and the color of the pixels under the object is 50,20,240. If the notSrcXOr transfer mode is used, the object's displayed color is 181,235,240.

The notSrcXOr mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear as though their ink were set to noOp.

null

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command, space constant

Summary

Equivalent to the null character (ASCII zero).

Syntax

Example Code

```
read from file modem: until null
```

Comments

Use the null constant as an easier-to-read substitute for numToChar(0).

Comments:

The null constant is needed because you can't type the character it represents in a script.

number

function

Synonyms

num

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

layer property, length function, number property

Summary

Returns the number of objects of a certain kind, or the number of chunks in a string.

Syntax

the number of [card|cd|background|bkgn|bg] {objectType | parts | controls}

the number of {objectType | parts | controls} [{in | of} group]

the number of {backgrounds | groups} [[{in | of} card] {in | of} stack]

the number of [marked] cards [{in | of} stack]

the number of {char[acters] | items | words | lines} {in | of} string

Example Code

```
repeat for the number of words in inputString  
if the number of card buttons > 0 then doTheButtons
```

Comments

Use the number function to find out how many of a thing there are.

Parameters:

The objectType is one of buttons (or btns), fields (or flds), images, graphics (or grcs), players, scrollbars, EPSs, audioClips, videoClips, groups, backgrounds, or cards.

The group is a reference to any group in the current stack.

The stack is any stack reference.

The string is any string or expression that evaluates to a string.

Value:

The number function returns a non-negative integer.

Comments:

If you use the form the number of card controls, the number function returns the number of controls on the current card that are not members of a group. The form the number of background controls returns the number of grouped controls that are in a group whose `backgroundBehavior` property is set to true.

The expression
the number of chars in string
is equivalent to
the length of string

The expression the number of backgrounds of stack returns the number of groups in the stack. The expression the number of groups of stack returns the number of groups on the current card of the stack. Groups that are nested inside another group are not counted.

Note: The number is implemented as a read-only global property, but can be more conveniently thought of as a function.

number

property

Synonyms
num

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
any keyword, first keyword, ID property, last keyword, layer property, middle keyword, name property, number function, sort command, About object types and object references, How to change the order of cards in a stack, How to refer to a control on another card

Summary
Specifies an object's position within a file, a card's position within a stack, or a control's layer on a card.

Syntax
set the number of card to number
get the number of object

Example Code
set the number of this card to 1 -- move to front of stack
if the number of this card is 1 then go last card
put the short number of last button into numberOfButtons

Comments
Use the number property to find out what layer a card is on, to find or change a card's position, or to refer to an object.

Value:
The number of an object is a non-negative integer.

For all objects except cards, the number property is read-only and cannot be set.

Comments:
The number of a control is its layer on the card. Lower numbers are further back; higher numbers are farther forward. If you create several controls and don't change their layer order, the first control you created has the number 1 and the rest of the controls are numbered in the order of creation.

The number of a card specifies its position in the stack. When you open a stack without specifying a card, the card whose number is one appears first. Setting the number of a card moves it to the specified position in the stack. (Cards are the only objects whose number property can be set.)

The number of a main stack is always zero. The number of a substack is its creation order in the file, from 1 to the number of substacks.

You can refer to any object by specifying its number.

numberFormat

property

Synonyms

Objects

local, scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

format function, round function, showValue property, trunc function

Summary

Specifies how many digits before and after the decimal point a computed number should have.

Syntax

set the numberFormat [of scrollbar] to formatExpression

Example Code

```
set the numberFormat to "#.00" -- dollar format
set the numberFormat of scrollbar "Progress" to "0.0"
```

Comments

Use the numberFormat property to specify the results of numeric calculations, or the display of numbers in scrollbars whose showValue property is set to true.

Value:

The numberFormat consists of any combination of a string of zeros and a string of hash marks (#), separated by a decimal point. (If the numberFormat contains hash marks, it must be surrounded by quotes; otherwise the first hash mark is interpreted as the start of a comment.)

By default, the numberFormat property is set to 0.#####. The numberFormat of newly created scrollbars is set to empty.

Comments:

The number of zeros or hash marks before the decimal point indicates how many digits are shown before the decimal point. If the result of a calculation has fewer digits before the decimal point than there are zeros or hash marks in the numberFormat, leading zeros are added. If the calculated result has more digits before the decimal point than there are zeros or hash marks, all the digits are displayed.

If there are zeros after the decimal point, the number of zeros indicates the number of digits after the decimal point in a calculated number. If there are more digits after the decimal point, the number is truncated. If there are fewer digits, trailing zeros are added.

If there are hash marks after the decimal point, the number of hash marks indicates the maximum number of digits after the decimal point. If there are more digits, the number is truncated, but if there are fewer digits, no trailing zeroes are added.

The default numberFormat is 0.#####, meaning that the results of calculations are reported with at least one digit before the decimal point, and up to 6 digits after the decimal point.

Important! Changing the numberFormat does not automatically change the format of a number that's already in a container. It affects numbers only when they are calculated and then displayed or used as strings. Otherwise, the number retains its full numeric precision.

Since the numberFormat is a local property, its value is reset to 0.##### when the current handler finishes executing. It retains its value only for the current handler, and setting it in one handler does not affect its value in other handlers it calls. (The numberFormat of a scrollbar is not reset in this way.)

Note: Since Revolution does not use decimal numbers for its internal calculations (for reasons of speed), the decimal representation of a number is sometimes slightly off the correct number. For example, 10^{-1} is equal to 0.1, but is calculated (to eighteen decimal places) as 0.100000000000000006. Because of this, setting the numberFormat to specify many decimal places after the decimal point may produce unexpected results in a statement that tests for an exact number. To prevent this, either avoid setting the numberFormat to a value more precise than you need, or use the abs function instead of the = operator to test equality:

```
set the numberformat to "#####"  
put  $10^{-1} = 1$  -- reports false because of the decimal error  
put  $\text{abs}((10^{-1}) - 1) = \text{zero}$  -- reports true
```

numeric

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

dateTime keyword, sort command, sort container command, text keyword

Summary

Used with the sort and sort container commands, to sort numbers as a unit, not as a sequence of characters.

Syntax

Example Code

```
sort cards numeric by field "Social Security Number"
```

Comments

Use the numeric keyword when sorting by a field or portion of a container that is a number.

Comments:

Alphabetical sorting does not take into account the special format of numbers with more than one digit. For example, a normal sort places 72 after 1280, because the first digit of 72 is greater than the first digit of 1280 (7 is greater than 1). The numeric keyword sorts numbers in numeric order, rather than alphabetical order.

numToChar

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

binaryDecode function, binaryEncode function, charSet property, charToNum function, ISOtoMac function, macToISO function, toLower function, toUpper function, uniEncode function, useUnicode property, Recipe for converting between characters and ASCII values

Summary

Returns the character corresponding to an ASCII value.

Syntax

the numToChar of ASCIIValue

numToChar(ASCIIValue)

Example Code

```
numToChar(65) -- returns A
```

```
numToChar(0) -- returns the null character, ASCII zero
```

Comments

Use the numToChar function to translate numbers into their character equivalents, or to interpret a character (such as a control character) that can't be displayed.

Parameters:

The ASCIIValue is an integer between zero and 255, or an expression that evaluates to such an integer. If the useUnicode property is set to true, the ASCIIValue is an integer between zero and 65535.

Value:

The numToChar function returns a single character.

Comments:

The numToChar function is the inverse of the charToNum function.

If the ASCIIValue is between 127 and 255, the character returned by the numToChar function depends on the character set currently in use. On Mac OS systems this is usually the Macintosh character set; on

Unix systems, this is usually ISO 8859; on Windows systems, this is usually Code Page 1252, a variant of ISO 8859.

If the `useUnicode` property is set to `true`, the `numToChar` function returns a double-byte character. If the `useUnicode` is `false` and you specify an `ASCIIValue` greater than 255, the `numToChar` function returns the character corresponding to the `ASCIIValue` mod 256.

Changes to Transcript:

The ability to handle Unicode characters was introduced in version 2.0. In previous versions, it was not possible for the `numToChar` function to return a Unicode character.

of
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

in keyword, owner property, the keyword

Summary

Designates the object a property belongs to, or the parameter of a function, or the string that contains a chunk expression.

Syntax

Example Code

```
set the backgroundColor of field "Alert" to red  
get the length of thisDate  
put word 12 of it after field "Go On"
```

Comments

Use the of keyword to designate an attribute of an object or chunk, or to designate a parameter of a built-in function.

Comments:

When designating a property, use the of keyword between the property name and the object reference. (The of keyword is not used when referring to global properties, only properties of objects.)

In an expression that includes a built-in function with one parameter, you can use either of two forms:

```
functionName(parameter)  
functionName of parameter
```

The of keyword cannot be used for custom functions, or for built-in functions with more than one parameter.

When designating a part of a string, use the of keyword after the chunk type. For example, to specify the first word of a string, use an expression like word 1 of string. In complex chunk expressions with

multiple chunk types, use the of keyword after each chunk type. For example, to specify the last character of the second line of a string, use an expression like this:

last char of line 2 of string

When designating the number of chunks in a string, use the of keyword after the chunk type. For example, to get the number of lines in a string, use an expression like this:

the number of lines of string

offset function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

itemOffset function, lineOffset function, wordOffset function

Summary

Returns the number of characters between the first character of a value and an occurrence of a specified string.

Syntax

`offset(charsToFind,stringToSearch[,charsToSkip])`

Example Code

```
offset("c","abcde") -- returns 3  
offset("b","abacadabra",2) -- returns 6  
offset("bark","embarking") -- returns 3
```

Comments

Use the offset function to find where a string occurs in another string.

Parameters:

The `charsToFind` is a string or an expression that evaluates to a string.

The `stringToSearch` is a string or an expression that evaluates to a string.

The `charsToSkip` is a non-negative integer. If you don't specify how many `charsToSkip`, the offset function does not skip any items and starts at the beginning of the `stringToSearch`.

Value:

The offset function returns a non-negative integer.

Comments:

The value returned by the offset function is the number of the character where `charsToFind` appears in `stringToSearch`. If the `charsToFind` is not in `stringToSearch`, the offset function returns zero.

If the `charsToFind` contains more than one character, and the entire `charsToFind` appears in the `stringToSearch`, the `offset` function returns the character number where the `charsToFind` starts.

If you specify how many `charsToSkip`, the `offset` function skips the specified number of characters in the `stringToSearch`. The value returned is relative to this starting point instead of the beginning of the `stringToSearch`.

on

control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

commandNames function, end keyword, function control structure, setProp control structure, About commands and functions, About the structure of a script, How to write your own commands and functions

Summary

Defines a message handler.

Syntax

```
on messageName [parametersList]
    statementList
end messageName
```

Example Code

Comments

Use the on control structure to handle a message, or to implement a custom command.

Form:

The first line of a message handler consists of the word "on" followed by the message's name. If the handler has any parameters, their names come after the message name, separated by commas.

The last line of a message handler consists of the word "end" followed by the handler's name.

Parameters:

The messageName is a string up to 65,535 characters in length. The messageName must not be a Transcript reserved word.

The parametersList consists of one or more parameter names, separated by commas.

The statementList consists of one or more Transcript statements.

Comments:

The purpose of a message handler is to perform an action, or actions, when the message is received.

A message handler can contain any set of Transcript statements.

The `messageName` is the name of the message to be handled. This can be either a built-in message (such as `mouseDown` or `openCard`) or a message that triggers a custom command handler. In either case, when a message called `messageName`, traversing the message path, arrives at an object, Revolution checks that object's script to see whether it has a message handler corresponding to the message. If it does, the statements in that message handler are executed.

You create a custom command by writing an `on` message handler for it. When you use the command in a script, a message corresponding to that command is passed through the message path. When it reaches the object whose script contains the `on` handler, the statements in the handler are executed.

one

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

arrow constant, constant command, first keyword

Summary

Equivalent to the number 1.

Syntax

Example Code

```
add one to field "Total" -- same as "add 1 to..."
```

Comments

Use the one constant when it is easier to read than the numeral 1.

onto
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

place command, remove command

Summary

Used with the place command to designate the card where a group is to be placed.

Syntax

Example Code

```
place background 1 onto this card
```

Comments

Use the onto keyword to designate a group to be placed on a card.

opaque
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

opaque property, style property, transparent keyword, Object menu > New Control > Blank Button

Summary

Used with the style property to indicate an opaque button or field.

Syntax

Example Code

```
set the style of button "Cancel" to standard
```

Comments

Use the opaque keyword to create a control that blocks the view of any object underneath it.

Comments:

The field or button is filled with its backgroundColor. If the field or button's backgroundColor is empty, it is filled with the backgroundColor of the object's owner.

Setting a control's style property to "opaque" sets its opaque property to true.

opaque property

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backgroundColor property, backgroundPattern property, blendLevel property, ink property, noOp keyword, visible property, Shortcut to make an image transparent

Summary

Specifies whether the inside of a control is opaque or transparent to the objects beneath it.

Syntax

set the opaque of object to {true | false}

Example Code

```
set the opaque of control 7 to false
```

Comments

Use the opaque property to change the appearance of an object.

Value:

The opaque of a control is true or false.

Comments:

If a control's opaque property is set to true, its entire rectangle is opaque, and objects underneath it cannot be seen.

If the opaque property is set to false, the text or other content and the borders of the object are drawn, but its background becomes transparent and objects beneath it show through.

If the object is an image, the setting of the opaque property has no effect. Painted areas of the image are opaque, and areas with no paint (transparent or "erased" areas) are not opaque, regardless of the setting of the image's opaque property.

Tip: Graphics whose ink property is set to noOp are transparent, but (unlike graphics whose opaque is set to false) they still intercept mouse clicks within the graphic's outline. To create an invisible hot spot with a non-rectangular shape, use a curve or polygon graphic and set its ink to noOp.

Setting a button's or field's style property to "opaque" sets its opaque to true.

Cross-platform note: The setting of a button's opaque property has no effect on Mac OS and OS X systems if the lookAndFeel is set to "Appearance Manager", the button's style is "standard", and the button's threeD and showBorder properties are both set to true. In this case, the button is drawn by the operating system's Appearance Manager routines and is always opaque, regardless of the setting of its opaque property.

open driver command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close driver command, COM1: keyword, driverNames function, modem: keyword, open file command, open process command, open socket command, printer: keyword, read from driver command, write to driver command, Why is there a problem with line endings?

Summary

Establishes a connection to a device so you can send data to it or get data from it.

Syntax

open driver driverName [for [text | binary] {update | read | write}]

Example Code

```
open driver "/dev/cu.modem" for binary write
open driver (line 3 of it) for text read
open driver "COM2:" -- equivalent to open file "COM2:"
```

Comments

Use the open driver command to communicate with USB devices, devices attached to a serial port other than the modem and printer port, and other peripheral devices.

Parameters:

The driverName specifies the name of a device driver that's installed on the system.

Comments:

If you don't specify binary or text mode, the driver is opened in text mode. (For most devices, you should use binary mode.)

Use the for read form to open the driver for reading. If the driver is opened for reading, you can use the read from driver command to get data from the device, but you can't send data to it.

Use the for write form to open the driver for writing. If the driver is opened for writing, you can use the write to driver command to send data to the device but you can't read from it.

Use the for update form to open the driver for both reading and writing. If the driver is opened for update, you can use both the read from driver and write to driver commands to send data to the device or get data from it.

If the driver does not exist, the result function is set to "Can't open that file." On OS X and Unix systems, you can obtain a list of available devices by reading the file `"/dev/tty"`.

Note: On Unix systems, devices can be addressed as part of the file system. This means that on such systems, the following two statements are equivalent:

```
open driver driverName  
open file "/dev/driverName"
```

Tip: Because OS X is based on Unix, you can use the open file command, as mentioned above, as a replacement for open driver on OS X systems.

On Windows systems, the open driver command is equivalent to the open file command when the driverName is COM1:, COM2:, etc.

If the device is a serial port, you can set the serialControlString property to specify the speed, parity, and other settings. Set the serialControlString before using the open driver command.

Changes to Transcript:

Support for using serial drivers with OS X systems was added in version 2.0.

open file command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

answer file command, ask file command, close file command, COM1: keyword, create folder command, files function, fileType property, LPT1: keyword, modem: keyword, openFiles function, open driver command, open process command, open socket command, printer: keyword, put command, read from file command, return constant, secureMode property, seek command, serialControlString property, sysError function, tempName function, URL keyword, write to file command, About filename specifications and file paths, How to associate files with a Mac OS standalone application, How to associate files with an OS X standalone application, How to associate files with a Windows standalone application, Why can't Revolution find a file I specified?, Why is there a problem with line endings?, Recipe for checking whether a volume or folder is write-protected

Summary

Opens a file so its contents can be accessed or modified.

Syntax

open file filePath [for [text | binary] {update | read | write | append}]

Example Code

```
open file "temp.txt"
open file "/etc/Glossary" for write
open file (myFilePath & "/" & myFileName) for binary read
```

Comments

Use the open file command to create a file or prepare an existing file for use by the read from file or write to file commands.

Parameters:

The filePath specifies the name and location of the file you want to open or create. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder. If the file you specify doesn't exist, Revolution creates it.

If you specify the name of a serial port on Mac OS or Windows systems, Revolution opens the specified port. The names of serial ports end in a colon (:).

Comments:

You can optionally specify either text or binary mode. If you specify text mode, when you use the write to file command to put data in the file, any line feed and return characters are translated to the appropriate end-of-line marker for the current operating system before being written to the file. (The end-of-line marker on Mac OS and OS X systems is a return character; on Unix, a line feed; on Windows, a CRLF.) When you use the read from file command to get data from the file, end-of-line markers are translated to the return constant, and any null characters are translated to spaces (ASCII 32). If you specify binary mode, null characters and end-of-line markers are not translated. If you don't specify a mode, the file is opened in text mode.

Use the for read form to open the file for reading. If the file is opened for reading, you can use the read from file command to examine its contents, but you can't modify it. (If you use the for read form and the file does not exist, Revolution does not create it, and the result function returns "Can't open that file.") Use this form for files on CD-ROM and other read-only media.

Use the for write form to open the file for writing. If the file is opened for writing, the write to file command replaces the file's contents from the starting point to the end of the file.

Use the for update form to open the file for both reading and writing. If the file is opened for update, you can use both the read from file and write to file commands to examine or change it, and writing to the file places the data at the specified position without replacing other characters beyond the written data.

Use the for append form to open the file for writing. If the file is opened for append, the write to file command adds its data to the end of the file without replacing its current contents.

If you don't specify a form, the file is opened for update.

On Mac OS and OS X systems, if the file doesn't already exist (so that the open file command creates it), the new file's creator signature and file type are set to the values in the fileType property.

Any files you have opened are closed automatically when you quit the application.

You can use the open file command to open a serial port on Mac OS or Windows systems. On Mac OS systems, specify either "printer:" or "modem:". On Windows systems, specify either "COM1:", "COM2:", or up to "COM9:". Set the serialControlString property before opening the port to specify the baud rate and other settings.

Tip: As an alternative to the open file, read from file, and write to file commands, you can also use the URL keyword with get, put, and other commands to access the contents of a file.

open printing command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

answer printer command, close printing command, print command, revShowPrintDialog command

Summary

Begins a batch print job, which allows more than one card to be printed on a page.

Syntax

open printing [with dialog [as sheet]]

Example Code

```
open printing
open printing with dialog
open printing with dialog as sheet
```

Comments

Use the open printing command to print multiple cards.

Comments:

Normally, the print command prints each card on a separate page. If you use the open printing command before issuing the print commands for the cards you want to print, the printing is delayed until all the cards have been specified. This causes the cards to be printed as one batch.

The open printing with dialog form opens the print dialog box on Mac OS systems. On Unix or Windows systems, this form acts like the open printing form, and no dialog box appears. If the as sheet form is used, the dialog box appears as a sheet on OS X systems.

If the open printing with dialog form is used, and the user cancels the print dialog box, the result is set to "Cancel".

The following handler prints cards that have been collected in a global variable:

```
on printSomeCards
    global cardsToPrint
```

```
open printing with dialog
if the result is "Cancel" then exit printSomeCards
repeat with x = 1 to the number of lines of cardsToPrint
  print card (line x of cardsToPrint)
end repeat
close printing -- send group of cards to printer
end printSomeCards
```

Changes to Transcript:

The open printing with dialog as sheet form was introduced in version 2.0.

open process

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close process command, hideConsoleWindows property, kill command, launch command, open file command, open driver command, open socket command, openProcesses function, read from process command, revGoURL command, secureMode property, shell function, write to process command, Why is there a problem with line endings?

Summary

Starts a program.

Syntax

open process appName [for [text|binary] {read | write | update | neither}]

Example Code

```
open process "/usr/bin/snapfile"  
open process it for read
```

Comments

Use the open process command to start up a process you want to either send data to or get data from or both.

Parameters:

The appName is the location and name of the program you want to open. (You can run only one instantiation of a given program at one time.) The appName can be any program on the system.

Comments:

Usually, you should use the open process command only to start up programs that run in the background, without any user interaction. When you are finished with the process, use the close process command to cause it to exit.

You can optionally specify either text or binary mode. If you specify text mode, when you use the write to process command, any line feed and return characters are translated to the appropriate end-of-line marker for the current operating system before being written to the process. (The end-of-line marker on Mac OS and OS X systems is a return character; on Unix, a line feed; on Windows, a CRLF.) If you

specify binary mode, return and line feed characters are not translated. If you don't specify a mode, the process is opened in text mode.

Use the read form to open the process for reading. If the process is opened for reading, you can use the read from process command to examine its output, but you can't send data to it.

Use the write form to open the process for writing. If the process is opened for writing, you can use the write to process command to send input data to it.

Use the update form to open the file for both reading and writing. If the file is opened for update, you can use both the read from process and write to process commands to send input to it or get output from it.

Use the neither form to run the process without reading data from it or sending data to it. (Using this form is equivalent to using the launch command to start up a program.) Processes opened with the neither form will quit automatically when finished running.

When you quit the application, processes opened with the read, write, or update forms are quit automatically.

Note: On OS X systems, you can use the open process command to start up an application, but not a Unix process. To work with a Unix process, use the shell function instead.

Changes to Transcript:

In versions before 1.1.1, when you quit the application, any processes that had been opened with the neither form were quit automatically on Windows systems.

Support for using the open process command on OS X systems was added in version 2.0.

open socket command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

accept command, close socket command, hostNameToAddress function, open file command, open driver command, open process command, openSockets function, read from socket command, resetAll command, socketError message, wait command, write to socket command

Summary

Establishes a TCP communications socket between your system and another system.

Syntax

open [datagram] socket [to] host[:port[[ID]]] [with message callbackMessage]

Example Code

```
open socket to "127.0.0.0:8080"  
open socket to "ftp.example.org:21|sendFiles"  
open socket to (field "Outgoing") with message "mailConnectionUp"
```

Comments

Use the open socket command to open a connection to another system on the Internet (or another IP network) to get and send data.

Parameters:

The host is the IP address or domain name of the host you want to connect to.

The port is the port number you want to connect to. If you don't specify a port, port 80 is used. (On most systems, port 80 is used for HTTP connections.)

The ID is an optional connection name for the socket, and can be any string. Specify an ID if you want to create more than one socket to the same host and port number, so that you can distinguish between them.

The callbackMessage is the name of a message to be sent when the connection is made.

Comments:

When a connection is made, the open socket command creates a new socket that can be used to communicate with the other system. The handler continues executing while the connection is being established. If you use the write to socket command while the connection is opening, the data is buffered and is sent to the host as soon as the connection is made.

Use the open datagram socket form if you want to send a connectionless UDP datagram instead of opening a socket.

If you specify a callbackMessage, the message is sent to the object whose script contains the open socket command, as soon as the connection is made. The first parameter of this message is the host and port number of the socket. Use a callbackMessage to trigger actions (such as reading from the socket) that must wait until the connection has been established. (To pause the handler that contains the open socket command until the callbackMessage is received, use the wait for messages form of the wait command.)

If the socket fails to open due to an error, a socketError message is sent to the object that attempted to open the socket. (If the error is due to a problem finding the specified host, the error message is returned in the result, and no socketError message is sent.)

Note: When the accept command creates a socket, it assigns a number as the connection name. If you are using both the open socket command and the accept command to connect to the same port on the same host, make sure to use a non-numeric connection name that won't conflict with the numbers assigned by the accept command. This ensures that you can always refer to two different sockets by distinct socket identifiers.

Important! Sockets are always opened in binary mode. This means that Revolution does not automatically convert between the other system's end-of-line convention and the line feed character (ASCII 10) that Revolution uses internally to mark the end of a line. If you are reading or writing data one line at a time, be sure you know whether the other system uses line feed, return (ASCII 13), or both to mark the end of each line; if necessary, you will need to convert end-of-line markers yourself, after receiving or before sending the data. (The usual end-of-line marker on Mac OS and OS X systems is a return character; on Unix, a line feed; on Windows, a CRLF.)

For technical information about the numbers used to designate standard ports, see the list of port numbers at <<http://www.iana.org/assignments/port-numbers>>, in particular the section titled "Well Known Port Numbers".

openBackground

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeBackground message, newBackground message, openCard message, openStack message, preOpenBackground message

Summary

Sent to the current card right after you go from a card that does not have the group to a card that does.

Syntax

openBackground backgroundID

Example Code

```
on openBackground theBackgroundID -- checkmark the current background
  put lineOffset(the short name of group ID theBackgroundID,
    button "Forms") into thisBgLineNumber -- button "Forms" is a menu
  put "!c" before line thisBgLineNumber of button "Forms"
end openBackground
```

Comments

Handle the openBackground message to change a group's objects, or perform other updates, when a card with the group on it is visited.

Parameters:

The backgroundID is the ID number of the background being opened.

Comments:

The openBackground message is sent to the card, then (if the card does not trap the message) along the message path to each group on the card. This means that if there is more than one group on the card, the openBackground message is sent to each one.

The openBackground message is sent after the destination card is visible. To make changes that take effect before the card becomes visible on the screen, use the preOpenBackground message instead.

The actual navigation is not triggered by the `openBackground` message, so trapping the message and not allowing it to pass does not prevent the card with the group from opening.

openCard

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

closeCard message, openBackground message, openStack message, preOpenCard message

Summary

Sent to a card right after you go to the card.

Syntax

openCard

Example Code

```
on openCard
  select text of field "Search"
end openCard
```

Comments

Handle the openCard message to change a card's objects, or perform other updates, when the card is visited.

Comments:

The openCard message is sent after the destination card is visible. To make changes that take effect before the card becomes visible on the screen, use the preOpenCard message instead.

The actual navigation is not triggered by the openCard message, so trapping the message and not allowing it to pass does not prevent the card from appearing.

openField

message

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeField message, exitField message, focusIn message, select command, selection keyword

Summary

Sent to an unlocked field when you click or select text in that field.

Syntax

openField

Example Code

```
on openField -- when clicking in the field, select all its text
    select text of the target
end openField
```

Comments

Handle the openField message if you want to do something when the user enters a field.

Comments:

The openField message is sent to buttons whose menuMode is "comboBox", since the type-in box in a combo box behaves like a field.

The openField message is not sent to locked fields. When a locked field becomes active (focused) or when text in it is selected by a handler, the focusIn message is sent to it.

openFiles

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close file command, open file command, openProcesses function, openSockets function, read from file command, write to file command

Summary

Returns a list of files that have been opened with the open file command, but not yet closed with the close file command.

Syntax

the openFiles
openFiles()

Example Code

```
the openFiles  
if it is not among the lines of the openFiles then open file it  
repeat until the openFiles is empty
```

Comments

Use the openFiles function to find out which files need to be closed, or to check whether a file is already open before reading from it or writing to it.

Value:

The openFiles function returns a list of file paths, one file per line.

Comments:

The list of file paths is in the same order that the files were opened.

Open files that were not opened by Revolution are not included in the list returned by the openFiles function.

openProcesses

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close process command, open process command, openFiles function, openProcessIDs function, openSockets function, processID function, there is a operator, there is no operator

Summary

Returns the names of processes that have been opened with the open process command, but have not yet exited.

Syntax

the openProcesses
openProcesses()

Example Code

```
the openProcesses  
put the openProcesses into processesToClose
```

Comments

Use the openProcesses function to find out which processes need to be closed, or to check whether a process is already running before communicating with it.

Value:

The openProcesses function returns a list of process names, one per line.

Comments:

The list of process names is in the same order used by openProcessIDs function, which is the same order that the processes were opened.

Programs that were not started by Revolution are not included in the list returned by the openProcesses function.

openProcessIDs

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

close process command, kill command, open process command, openFiles function, openProcesses function, openSockets function, processID function, signal message, there is a operator

Summary

Returns the process IDs of processes that have been opened with the open process command, but have not yet exited.

Syntax

the openProcessIDs

openProcessIDs()

Example Code

```
the openProcessIDs
put last line of the openProcessIDs into previousProcessID
```

Comments

Use the openProcessIDs function to communicate with processes you have opened with the open process command.

Value:

The openProcessIDs function returns a list of integers, one per line.

Comments:

Certain Unix commands such as "kill" require a process ID. You can use these commands to act on processes you opened by using the shell command:

```
put "kill -9" && line 1 of the openProcessIDs into whatToExecute
get shell(whatToExecute)
```

On Mac OS and OS X systems, there is no process ID. Instead, the openProcessIDs function returns a list of integers. The first application started up (with the open process or launch command) during a session is assigned the number 1, the second is assigned the number 2, and so on. After an application

quits, its process number is not re-used, so you can use the `openProcessIDs` function to determine how many times Revolution has started up an application during the current session.

The list of process IDs is in the same order used by `openProcesses` function, which is the same order that the processes were opened.

Programs that were not started by Revolution are not included in the list returned by the `openProcessIDs` function.

openSockets

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

accept command, close socket command, open socket command, openFiles function, openProcesses function, resetAll command

Summary

Returns a list of the currently open sockets.

Syntax

the openSockets
openSockets()

Example Code

```
the openSockets  
if line thisLine of the openSockets is "example.com:80" then beep
```

Comments

Use the openSockets function to find out which sockets need to be closed, or to check whether a socket is already open before reading from it or writing to it.

Value:

The openSockets function returns a list of the open sockets, one per line.

Comments:

Each line of the list returned by the openSockets function is a socket identifier. A socket identifier consists of the host and port number the socket is connected to, separated by a colon. If a connection name or number was assigned when the socket was opened, it is appended to the identifier, separated from the port number by a vertical bar (|).

Note: Several of the commands and functions in the Internet library use sockets, so the openSockets function may return sockets opened by the Internet library in addition to any sockets you have opened with the open socket command.

openStack

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

openBackground message, openCard message, preOpenStack message, resumeStack message, How to respond when a stack opens, Why does my stack open slowly?

Summary

Sent to the destination card right after you open a stack.

Syntax

openStack

Example Code

```
on openStack
  send "rollCredits" to field "Credits"
end openStack
```

Comments

Handle the openStack message to change a stack's objects, or perform other updates, when the stack is opened.

Comments:

The openStack message is sent after the stack is visible.

To make changes that take effect before the stack becomes visible on the screen, use a preOpenStack handler instead of openStack.

The actual opening is not triggered by the openStack message, so trapping the message and not allowing it to pass does not prevent the stack from opening.

openStacks

function

Synonyms
windows

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

defaultStack property, go command, name property, revLoadedStacks function, stacks function, How to find out whether a window is open, Why does Revolution ask to purge a stack?, Recipe for finding out whether a window is open by title, Window menu

Summary

Returns a list of the names of all open stacks.

Syntax

the openStacks
openStacks()

Example Code

```
the openStacks  
if it is not among the lines of the openStacks then open stack it
```

Comments

Use the openStacks function to determine which stacks are open.

Value:

The openStacks function returns a list with the short name property of each open stack, one per line.

Comments:

The list includes stacks that are part of the Revolution development environment (such as the message box and menu bar).

The list also includes open stacks that cannot be seen because their visible property is set to false.

The order of the list is determined by the order of the windows on the screen. The frontmost window is first.

option command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

menuButton function, menuMode property, menuName property, popup command, pulldown command, style property

Summary

Displays an option menu whose menu items are buttons in a stack.

Syntax

option stack

Example Code

```
option stack "Hot Text Items"  
option stack (the menuStack of this image) -- custom property
```

Comments

Use the option command to display a stack menu as an option menu in situations where it is not possible to attach the menu to a button (for example, if the contents of the menu are not known in advance).

Parameters:

The stack is any stack reference. The stack's first card must contain a button for each menu item in the option menu.

Comments:

Use the option command in a mouseDown handler to display the menu:

```
on mouseDown  
    option stack "Help Menu"  
end mouseDown
```

While the menu is displayed, the handler pauses.

The menu appears centered over the center of the control containing the handler. Choosing a menu item from the menu sends a mouseUp message to the stack.

option

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

cascade keyword, comboBox keyword, menuMode property, popup keyword, pulldown keyword, style property, tabbed keyword, Object menu > New Control > Option Menu

Summary

Specifies one of the menu types that can be used with the menuMode property.

Syntax

Example Code

```
set the menuMode of button "Style" to option
```

Comments

Use the option keyword to create a popup, option, or drop-down menu.

Comments:

If the lookAndFeel property is set to "Appearance Manager" or "Macintosh", the option keyword designates a popup menu. If the lookAndFeel is "Motif", the option keyword designates an option menu. If the lookAndFeel is "Windows 95", the option keyword designates a drop-down list similar to a combo box.

Option menus are normally used to designate a state. Choosing an item from an option menu changes the menu's current setting, making this menu type suitable for situations in which the user must choose one of the predefined states.

optionKey

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

altKey function, commandKey function, controlKey function, down constant, keyDown message, keysDown function, keyUp message, metaKey function, optionKeyDown message, shiftKey function, up constant

Summary

Returns the state of the Option key.

Syntax

the optionKey
optionKey()

Example Code

```
wait until the optionKey is down  
put the optionKey into item 2 of keyState
```

Comments

Use the optionKey function to check whether the Option key, Meta key, or Alt key is being pressed. You can use optionKey to add alternative capabilities to user actions such as clicking.

Value:

The optionKey function returns down if the key is pressed and up if it's not.

Comments:

The altKey, optionKey, and metaKey functions all return the same value. Which one to use is a matter of preference.

The terminology varies depending on platform. Users of different operating systems may know this key as the Option key (Mac OS systems), Meta key (Unix systems), or Alt key (Windows systems).

optionKeyDown

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

altKey function, commandKeyDown message, controlKeyDown message, keyDown message, metaKey function, optionKey function, rawKeyDown message

Summary

Sent when the user presses an Option key, Meta key, or Alt key combination.

Syntax

optionKeyDown keyName

Example Code

```
on optionKeyDown theKey -- strip high bit
  answer numToChar(charToNum(theKey) - 128)
end optionKeyDown
```

Comments

Handle the optionKeyDown message if you want to provide Option key shortcuts, or do something special when the user types an Option key combination.

Parameters:

The keyName is the actual character typed.

Comments:

The optionKeyDown message is sent only when the user types a key combination that produces a character. For example, typing Option-F11 does not send an optionKeyDown message, because Option-F11 does not produce a character.

The message is sent to the active (focused) control, or to the current card if no control is focused.

The terminology varies depending on platform. The optionKeyDown message is sent when the user types a character while holding down the Option key (Mac OS systems), Meta key (Unix systems), or Alt key (Windows systems).

or
operator

Synonyms

Objects
logical

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

and operator, bitOr operator, bitXOr operator, not operator, Operator Precedence Reference

Summary

Evaluates to true if either operand is true, false if both operands are false.

Syntax

value1 or value2

Example Code

```
("a" > "b") or ("b" > "a") -- evaluates to true  
(1 < 0) or (1 = 0) -- evaluates to false  
wait until the controlKey is up or the commandKey is up
```

Comments

Use the or operator to combine two or more logical (true or false) values.

Parameters:

The value1 and value2 are true or false, or expressions that evaluate to true or false.

Comments:

If value1 is true or value2 is true, or if both value1 and value2 are true, then the or operation evaluates to true. Only if both value1 and value2 are false does the expression value1 or value2 evaluate to false.

You can combine the logical operators and, or, and not in an expression.

Note: Transcript uses what is known as "short-circuit evaluation" for logical operators. This means that value1 is evaluated first. If value1 is true, the expression value1 or value2 is true regardless of what value2 is (because the expression evaluates to true as long as at least one of the values is true). In this case, Revolution does not evaluate value2, since doing so is not necessary to determine the value of value1 or value2. For example, evaluating the expression asin(2) normally causes an execution error (because 2 is not a legal argument for the arc sine function), but evaluating the expression (1 = 1) or

$(\text{asin}(2) = 1)$ does not cause an error: since $(1 = 1)$ is always true, the whole statement is always true and Revolution never tries to evaluate the asin function.

orientation

property

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

height property, rectangle property, width property

Summary

Reports whether a scrollbar is vertical or horizontal—that is, whether its height or width property is greater.

Syntax

get the orientation of scrollbar

Example Code

```
if the orientation of scrollbar 2 is vertical then beep
```

Comments

Use the orientation property to find out whether a scrollbar is horizontal or vertical.

Value:

The orientation of a scrollbar is "vertical" or "horizontal".

This property is read-only and cannot be set.

Comments:

If the scrollbar's height is greater than or equal to its width, the scrollbar is vertical. If the width is greater than the height, the scrollbar is horizontal.

To change a scrollbar's orientation, change its width and height properties.

oval

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

brushColor property, brushPattern property, choose command, lineSize property, penColor property, tool function, Object menu > New Control > Oval Graphic

Summary

Designates the paint tool used to draw ovals and circles. It also specifies, through the style property, that a graphic object is an oval shape.

Syntax

Example Code

```
set the style of the templateGraphic to oval  
choose oval tool
```

Comments

Use the oval keyword to paint an oval or circle with the penColor or to change a graphic to an oval shape.

Comments:

When using the Oval tool, the cursor is a crosshairs shape (over images) or an arrow (anywhere else). This tool is used to draw an oval or circle in the penColor, filled with the brushColor (or brushPattern).

If you try to paint with the Oval tool on a card that has no images, an image the size of the card is created to hold the painting. If the current card already has one or more images, painting outside the image has no effect.

Setting the style of a graphic to oval makes the graphic into an oval. Oval graphics, unlike painted ovals, can be changed and reshaped: use the height and width properties to change the shape and size of the oval.

owner
property

Synonyms

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
abbreviated keyword, effective keyword, ID property, long keyword, mainStack property, name property, number property, short keyword, About messages and the message path, How to include an object in the message path of every object, How to make an object the same color as its owner

Summary
Reports which object is next in the object hierarchy.

Syntax
get the [long | abbr[ev[iated]] | short] owner of object

Example Code
`if the owner of me is not the name of card 1 then go to previous card`

Comments
Use the owner property to find out which object another object belongs to in the object hierarchy.

Value:
The owner of an object is a string, which is the name property of the owning object.

The owner property is read-only and cannot be set.

Comments:
If the owner's name is empty, the ID is reported instead of the name.

The long, abbreviated, and short owner formats are described in detail in the entries for the name and ID properties. If you don't specify a format, the abbreviated owner is reported.

The owner of a main stack is empty.

The owner of a substack is its main stack.

The owner of a card is the stack it resides in.

The owner of a grouped control is its group.

The owner of a card control is the card it resides on.

The owner of an audio clip or video clip is the stack it resides in.

Changes to Transcript:

The ability to specify the long owner, short owner, and abbreviated owner forms was introduced in version 2.0. In previous versions, the abbreviated owner form was always used.

pageCount

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

currentPage property, postScript property, prolog property

Summary

Reports how many pages are in the PostScript code of an EPS object.

Syntax

get the pageCount of EPSObject

Example Code

```
add the pageCount of EPS "Watermark" to totalPages  
answer the pageCount of EPS 1 && "pages to print."
```

Comments

Use the pageCount property to find out how many pages are in an EPS object.

Value:

The pageCount of an EPS object is a positive integer.

This property is read-only and cannot be set.

Comments:

This property is supported only on Unix systems with Display PostScript installed.

pageHeights

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

borderWidth property, formatForPrinting property, formattedHeight property, height property, pageHeights property, textHeight property, textHeightSum function

Summary

Reports the height of each printed age if a field is printed.

Syntax

get the pageHeights of field

Example Code

```
set the height of field 1 to line 1 of the pageHeights of field 1
```

Comments

Use the pageHeights property to determine how much to scroll a field during printing.

Value:

The pageHeights of a field is a list of integers, one per line.

This property is read-only and cannot be set.

Comments:

The value reported by the pageHeights property is a list of numbers separated by returns. Each number is the height in pixels of a page full of text.

You can use the pageHeights property to print the entire contents of a field by printing the field, setting the field's scroll to the first line of the pageHeights, printing the field again, setting the scroll to the current scroll plus line 2 of the pageHeights, and so on.

The computations used by the pageHeights property assume the field's borderWidth property is set to zero and its margins is set to 6.

pageIncrement

property

Synonyms
pageInc

Objects
scrollbar

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
lineIncrement property, repeatRate property, scrollbarPageDec message, scrollbarPageInc message, thumbPosition property

Summary
Specifies how far a scrollbar moves when the user clicks somewhere in its gray region.

Syntax
set the pageIncrement of scrollbar to pageAmount

Example Code
set the pageIncrement of scrollbar "Content" to 200

Comments
Use the pageIncrement property to change the amount that is scrolled when the gray region is clicked.

Value:
The pageIncrement of a scrollbar is a number between the scrollbar's startValue and endValue.

By default, the pageIncrement property of newly created scrollbars is set to 8192.

Comments:
When the user clicks in the gray region above or below the scrolling thumb (to the left or right, for a horizontal scrollbar), the scrollbar moves one page up or down (or to the left or right). Use the pageIncrement property to specify how far the scrolling thumb moves.

The startValue and endValue properties set the scale used for the pageIncrement, so for example, if the pageIncrement is set to one-tenth of the difference between the startValue and endValue, clicking the gray region moves the scrollbar one-tenth of its length.

paint
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

export command, import command

Summary

Used with the import and export commands to specify that the file to be imported or exported is in a bitmapped format.

Syntax

Example Code

```
import paint from file it
```

Comments

Use the paint keyword to indicate that you are importing or exporting a bitmapped image.

Comments:

When the paint keyword is used with the export command, the specified image is exported as a PBM file.

When the paint keyword is used with the import command, a GIF, JPEG, PNG, BMP, XWD, XBM, XPM, or PBM, PGM, or PBM file can be imported. On Mac OS systems, PICT files can also be imported (but they cannot be displayed on Unix or Windows systems).

paintCompression

property

Synonyms

Objects

global, image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

choose command, export command, import command, import snapshot command, JPEGQuality property, Image Types Reference, File menu > Import As Control > Image File..., Object menu > New Control > Image

Summary

Specifies the compression format used for storing an image.

Syntax

get the paintCompression of image

set the paintCompression to {png | jpeg | gif | rle}

Example Code

```
set the paintCompression to "PNG"  
if the paintCompression of image "Photo" is "pict" then warnUser
```

Comments

Use the paintCompression property to obtain the format of images, or to change the format used for new images.

Value:

The paintCompression is one of the following: "png", "jpeg", "gif", or "rle". By default, the global paintCompression property is set to "rle".

The paintCompression of an image is one of the following: "png", "jpeg", "gif", "rle", or "pict". By default, the paintCompression property of a newly created image is set to "rle" if it was created with the create command or by using a paint tool. If the image was created with the import command, its paintCompression is set to the format of the imported picture file.

For images, this property is read-only and cannot be set; you can set only the global paintCompression.

Comments:

When an image is changed with a paint tool, it is recompressed the next time you leave the card it's on. The format in which it is compressed is set by the global `paintCompression` property.

To change an image's compression format, first set the `paintCompression` to the desired value, then paint in the image. Then either go to another card and return, or close and re-open the stack.

Cross-platform note: On Mac OS and OS X systems, PICT files can be imported, and the `paintCompression` of the resulting image is set to `pict`. However, PICT images cannot be displayed on Unix or Windows systems unless they are converted to another format.

To compress an image in GIF format, you must purchase a special license key. (Contact support@runrev.com for information.) If you set the `paintCompression` to `"gif"` and then recompress an image, and you don't have this special license key, the image is compressed in RLE format instead.

If an image's `alphaData` property contains any value other than 255 (opaque), it is automatically recompressed in PNG format to preserve the alpha channel data.

palette

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

activatePalettes property, go command, hidePalettes property, modal command, mode property, modeless command, raisePalettes property, sheet command, style property, topLevel command, About windows, palettes, and dialog boxes, How to keep a window on top of other windows, Why does a stack window open in the wrong mode?

Summary

Opens a stack in a palette window.

Syntax

palette stack

Example Code

```
palette stack "Tools"  
palette the defaultStack
```

Comments

Use the palette command to display a stack in a palette window.

Parameters:

The stack is any stack reference.

Comments:

Use palette windows to display tools or information about main windows.

A palette window behaves like an ordinary window, except that its appearance is different, with a narrower title bar. If the raisePalettes property is set to true, any palettes float in their own layer, above all ordinary windows. Palettes cannot be resized or edited. To edit a palette, use the topLevel command to display it in an editable window.

The palette command closes the stack and reopens it as a palette, so closeStack and openStack, closeCard and openCard, and (if applicable) closeBackground and openBackground messages are sent to the current card as a result of executing this command. Use the lock messages command before

executing palette if you want to prevent the close messages from being sent; the open messages are sent regardless of the setting of the lockMessages property.

If the stack is already displayed as a palette, the palette command does not close and reopen it.

The Browse tool is always used in palette windows, regardless of the current setting of the tool property.

palindromeFrames

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentFrame property, frameCount property, frameRate property, repeatCount property

Summary

Specifies that a looping animated GIF plays forward and backward.

Syntax

set the palindromeFrames of image to {true | false}

Example Code

```
set the palindromeFrames of image "Construction" to true
```

Comments

Use the palindromeFrames property to change the way an animated GIF image repeats itself.

Value:

The palindromeFrames of an image is true or false.

By default, the palindromeFrames property of newly created images is set to false.

Comments:

An animated GIF image can be set to loop over and over, instead of playing the animation once and then stopping. If the palindromeFrames property is set to true, the image plays forward, then plays backward from the end, then plays forward again, and so on.

If the palindromeFrames is false, the image plays from the beginning to the end, then skips back to the beginning to start the next loop.

If the contents of the image is not an animated GIF, or the image's repeatCount is zero, the setting of the palindromeFrames property has no effect.

pan property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

nodes property, constraints property, tilt property, zoom property

Summary

Specifies the current horizontal view angle of a QuickTime VR movie.

Syntax

set the pan of player to degrees

Example Code

```
set the pan of player "Arctic" to 90.5  
put the pan of player myPlayerName into myLocation
```

Comments

Use the pan property to find out where the user is in a QuickTime VR movie.

Value:

The pan is a number between zero and 360.

Comments:

The user can move the view of a QuickTime VR movie using the navigational controls in the player; a handler can change the view by setting the player's pan and tilt properties.

The pan specifies the amount of rotation in the horizontal plane, in degrees. (Think of a person standing in the middle of a scene and turning from side to side. The point where the person is standing is the `currentNode`, and the amount of turning is the pan.) A pan of zero corresponds to the straight-ahead view of the scene. As the viewer turns clockwise, the pan increases.

If you set the pan of a player to a number outside the range zero to 360, no error results, but the pan is set to number mod 360. For example, if you attempt to set the pan of a player to -20, its pan is actually set to 340.

The pan is limited by the player's constraints property. If you specify a pan greater than the range permitted by the constraints, the pan is set to the highest permitted value. If you specify a pan less than the range permitted by the constraints, the pan is set to the lowest permitted value.

If the player does not contain a QuickTime VR movie, its pan property is zero.

param function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

function control structure, getProp control structure, on control structure, paramCount function, params function, setProp control structure, variableNames function

Summary

Returns the specified parameter passed to the current handler.

Syntax

the param of parameterNumber

param(parameterNumber)

Example Code

```
param(3) -- returns the third parameter  
param(0) -- returns the handler name
```

Comments

Use the param function within a handler to get the value of a parameter when you don't know in advance how many parameters will be passed to the handler.

Parameters:

The parameterNumber is a non-negative integer.

Value:

The param function returns the parameter value specified by the parameterNumber. If the parameterNumber is zero, the param function returns the handler name.

Comments:

Usually, you assign names to parameters in the first line of a function handler or message handler. For example, the following function assigns three parameters, which are multiplied together:

```
function product firstFactor,secondFactor,thirdFactor  
  return firstFactor * secondFactor * thirdFactor  
end product
```

But if you want to multiply all the numbers passed to the function handler together, you don't know ahead of time how many parameters will be passed, so you can't assign a parameter name (such as firstFactor) to each one in the first line of the function handler. In this case, you can use the param function to use each parameter without needing to assign it a name:

```
function product
  put 1 into total
  repeat with nextFactor = 1 to the paramCount
    multiply total by param(nextFactor)
  end repeat
  return total
end product
```

Revolution evaluates the parameters before passing them. So if you call myHandler with the following statement:

```
myHandler 1+1,"A","Hello" && "world"
```

the parameters returned by the param function are 2, A, and "Hello World".

paramCount

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

function control structure, getProp control structure, on control structure, param function, params function, setProp control structure, variableNames function

Summary

Returns the number of parameters passed to the current handler.

Syntax

the paramCount
paramCount()

Example Code

```
the paramCount  
repeat with thisParam = 1 to the paramCount
```

Comments

Use the paramCount function to find out how many parameters were passed to the current handler.

Value:

The paramCount function returns a non-negative integer.

Comments:

The paramCount function returns the number of parameters passed to the current handler. If no parameters were passed, the paramCount function returns zero.

params function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

function control structure, getProp control structure, on control structure, param function, paramCount function, setProp control structure, variableNames function

Summary

Returns all the parameters passed to the current handler.

Syntax

the params
params()

Example Code

```
the params  
put char 2 to -2 of word 1 of the params into handlerName
```

Comments

Use the params function within a handler to get the parameters that were passed to the handler.

Value:

When used in a message handler, the params function returns the handler name, a space, and a list of parameter values, each enclosed in quotes and separated by commas.

When used in a function handler, the params function returns the handler name, then the list of parameter values in parentheses. Each parameter value is enclosed in quotes and separated by commas.

Comments:

Usually, you assign names to parameters in the first line of a function handler or message handler. For example, the following handler assigns three **Parameters**:

```
on myHandler thisParam,thatParam,theOtherParam  
  answer thisParam & return & thatParam & return & theOtherParam  
end myHandler
```

If you call the above handler with four parameters, the first three parameters are assigned to the names `thisParam`, `thatParam`, and `theOtherParam`, but the fourth parameter is not assigned a name:

```
myHandler "red","green","blue","yellow"
```

You can obtain the fourth parameter for use in the handler with the `params` function:

```
on myHandler thisParam,thatParam,theOtherParam
  put item 4 of the params into yetAnotherParam
  answer yetAnotherParam
end myHandler
```

In this case, item 4 of the `params` is "yellow". (To use the value itself, you need to remove the opening and closing quotes.)

If the `params` function is used in a function handler, the parameters are enclosed in parentheses. For example, the following function handler has three **Parameters**:

```
function myFunction thisParam,thatParam,theOtherParam
  return thisParam & return & thatParam & return & theOtherParam
end myFunction
```

If you call "myFunction" with the following statement:

```
get myFunction("red","green","blue")
the value returned is "myFunction("red","green","blue")".
```

Revolution evaluates the parameters before passing them. So if you call `myHandler` with the following statement:

```
myHandler 1+1,"A","Hello" && "world"
the value returned by the params function is
myHandler "2","A","Hello world"
```

Changes to Transcript:

The format of the `params` when used in a function handler was changed in version 2.0. In previous versions, the format for functions was the same as the format for message handlers: the parameters were not enclosed in parentheses, but instead separated from the handler name by a space.

pass
control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

dynamicPaths property, end keyword, exit control structure, function control structure, getProp control structure, next repeat control structure, on control structure, send command, setProp control structure, start using command, Recipe for switching between styled text and HTML source

Summary

Stops the current handler and passes the message to the next object in the message path.

Syntax

pass messageName [to top]

Example Code

Comments

Use the pass control structure to end a handler while letting the message continue along the message path.

Form:

The pass statement appears on a line by itself, anywhere inside a handler.

Parameters:

The handler is the name of the handler in which the pass control structure appears.

Comments:

When the pass control structure is executed, any remaining statements in the handler are skipped. Hence, the pass control structure is usually used either at the end of a handler or within an if control structure.

Use the pass control structure at the end of a handler to make sure that objects further up the message path, or Revolution itself, receive the message. For example, if a stack's script contains a closeCard handler that does housekeeping tasks, and a particular card needs to perform additional tasks if the stack is closed when on that card, the card's closeCard handler can perform those additional tasks, and then

use the pass control structure to let the closeCard handler in the stack's script receive the message and be executed. The following example demonstrates the idea:

```
on closeCard -- in card script
  put empty into field "Search"
  pass closeCard -- give stack script a crack at it
end closeCard
```

Built-in messages that perform a task, such as keyDown and closeStackRequest, must be received by the engine or the task will not be performed. For example, Revolution enters a typed character into a field when it receives the keyDown message, and starts closing a stack when it receives the closeStackRequest message. For this reason, if you create a handler for a built-in message that performs a task, make sure to use the pass control structure to ensure that the engine receives the message after the handler is finished with it.

Similarly, if you set a custom property, the setProp trigger must be received by the engine, or the custom property will not be set. This means that if you create a setProp handler to intercept requests to set a custom property, the property is not set unless you include a pass control structure in the setProp handler.

When a handler executes a pass statement, the message is passed to the next object in the message path. If you use the pass...to top form of the pass control structure, the message is passed directly to the engine, without being passed through any other object in the message path.

To halt the current handler without passing the message on through the message path, use the exit control structure instead. To halt the current handler and return a result, use the return control structure instead.

Important! You cannot use the pass command to pass a message that was originally sent with the send command.

Changes to Transcript:

The pass...to top form was added in version 2.1.

passKey

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

password property

Summary

Enters a password for a locked stack, letting the user modify the stack.

Syntax

set the passKey of stack to passwordString

Example Code

```
set the passKey of this stack to field "Challenge"
```

Comments

Use the passKey property to unlock a password-protected stack.

Value:

The passKey of a stack is a string.

By default, the passKey property of newly created stacks is set to empty.

Comments:

If a stack has been locked against modification with a password, you can set the passKey property of the stack to the correct password to unlock the stack. Setting the passKey property is the script equivalent of typing the password into Revolution's password dialog box. Use this property if you want to perform password authentication in a handler.

password

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

passKey property, How to use a password for an FTP server, How to use a password for a web page, Why is a custom property garbled in a protected stack?

Summary

Specifies a password that the user must enter before making changes to the stack.

Syntax

set the password of stack to {passwordString | empty}

Example Code

```
set the password of stack "Secrets" to field "Password"  
set the password of this stack to empty -- remove password
```

Comments

Use the password property to encrypt a stack's contents, allowing access only from within the application.

Value:

The password of a stack is a string.

By default, the password property of newly created stacks is set to empty.

Comments:

If the password property of a stack is not empty, all the text in the stack is encrypted (so that it cannot be read in another program, such as a text editor). Scripts, custom properties, text in fields or buttons, and object names in a password-protected stack are all encrypted. However, you can still open the stack, see the contents, and get object properties.

The password property applies to a stack, not to the entire stack file, so it is possible to have a stack file that contains both password-protected and unprotected stacks.

If the password is set, the stack's scripts cannot be modified, its password cannot be changed, and objects cannot be copied.

These restrictions stop operating temporarily for the current session when the stack's passKey property is set to the correct password by a handler.

To permanently remove the password restrictions, set the stack's password to empty. Setting the password to empty removes the password from the stack and makes it fully accessible to any user.

Note: Since all the text in a password-protected stack must be decrypted when the stack is opened, a password-protected stack takes longer to open than an unencrypted one, especially if the stack is large.

The password is encrypted when stored in the stack to prevent cracking the password protection. If you get the password of a stack, the property reports the encrypted text of the password, rather than the password itself.

Important! Password-protected stacks may cause some problems when opened in the Revolution development environment. (For example, the Application Browser window cannot display properties of a password-protected stack.) If you want to set a password for stacks before you release them, the recommended method is to set the password in Step 3 of the Distribution Builder when you package the final application for distribution.

paste

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clipboard function, copy command, cut command, newButton message, newCard message, newEPS message, newField message, newGraphic message, newGroup message, newImage message, newPlayer message, newScrollbar message, undo command, How to copy a card, How to move a card between stacks, Why is a control the wrong size when created by a handler?, Edit menu > Paste

Summary

Pastes the contents of the clipboard into the selection or insertion point.

Syntax

paste

Example Code

```
paste
if the clipboard is "text" then paste
```

Comments

Use the paste command to place objects, an image, or text on the current card.

Comments:

If the clipboard contains one or more controls, they are placed on the current card. If an object is being pasted, the paste command places the ID property of the newly created object in the it variable, and the appropriate message is sent.

If the clipboard contains text, it is placed in the current field. If there is no insertion point in any field, the paste command does nothing.

If the clipboard contains graphic data, it is placed in the current image. If there is no current image, the paste command does nothing.

If the clipboard contains a card, the card is pasted into the defaultStack, after the current card.

The paste command is equivalent to choosing Edit menu Paste.

pasteKey

message

Synonyms

Objects

field, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

copyKey message, cutKey message, undoKey message

Summary

Sent when the user presses the key combination equivalent to the "Paste" menu item.

Syntax

pasteKey

Example Code

```
on pasteKey -- replace returns with spaces before pasting
  if the clipboardData["text"] is not empty then set

    the clipboardData["text"] to

      replaceText(the clipboardData["text"],return,space)
  paste
end pasteKey
```

Comments

Handle the pasteKey message if you want to change the normal pasting process, or prevent use of the Paste keyboard equivalent without changing the menu.

Comments:

The Revolution development environment traps the pasteKey message, unless "Suspend Revolution UI" is turned on in the Development menu. This means that the pasteKey message is not received by a stack if it's running in the development environment.

The pasteKey message is sent when the user presses Command-V (on Mac OS systems), Control-V (on Windows systems), Shift-Insert (on Unix systems), or the keyboard Paste key.

patterns

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundPattern property, borderPattern property, bottomPattern property, colors property, focusPattern property, foregroundPattern property, hilitePattern property, shadowPattern property, topPattern property, Development menu > Image Library

Summary

Specifies all the patterns of an object, in shorthand form.

Syntax

set the patterns of object to patternsList

Example Code

```
set the patterns of button 1 to the patterns of card "Template"  
set the patterns of me to storedPatternsList
```

Comments

Use the patterns property to get all eight basic pattern properties at once, or to set the patterns of one object to be the same as the patterns of another object.

Value:

The patterns of an object is a list of pattern specifiers, one per line. A pattern specifier is a built-in pattern number between 1 and 164 (corresponding to Revolution's built-in patterns 136 to 300), or the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

The patterns of an object contains eight lines, some of which may be empty.

Comments:

You can set all these patterns individually; the patterns property simply provides a shorter method of dealing with all of them at once. Each line of the patterns corresponds to one of the following pattern properties:

ī Line 1: the foregroundPattern

- ï Line 2: the backgroundPattern
- ï Line 3: the hilitePattern
- ï Line 4: the borderPattern
- ï Line 5: the topPattern
- ï Line 6: the bottomPattern
- ï Line 7: the shadowPattern
- ï Line 8: the focusPattern

If you leave a line blank when setting the patterns, the pattern property corresponding to that line is left unchanged.

If the patterns property of an object reports a blank line, the corresponding pattern is not set for the individual object, but is inherited from the object's owner. Use the form the effective patterns of object to obtain the patterns used for the object, whether set for the object or inherited.

If a pattern is set for an object, that pattern is used instead of the corresponding color for that object.

paused

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentTime property, duration property, looping property, play command, revGoToFramePaused command, start command, stop command

Summary

Specifies whether a sound or movie is paused.

Syntax

set the paused of player to {true | false}

Example Code

```
set the paused of player "Oops!" to true -- pause the movie
```

Comments

Use the paused property to pause or resume a movie or sound, or to check whether the user has done so.

Value:

The paused of a player is true or false.

Comments:

When a player is not playing, its paused reports true.

peerAddress

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

accept command, hostAddress function, hostAddressToName function, open socket command, openSockets function

Summary

Returns the IP address of the remote system at the other end of a socket.

Syntax

the peerAddress of host:port[|connectionID]
peerAddress(host:port[|connectionID])

Example Code

```
peerAddress("ftp.example.org:21")  
the peerAddress of "www.example.org:8080|primaryConnection"  
the peerAddress of mySocket
```

Comments

Use the peerAddress function to find the Internet address of the computer a socket is connected to.

Parameters:

The host is an IP address or domain name.

The port is the port number of the port the socket is connected to.

The connectionID is a string identifying the socket.

Value:

The peerAddress function returns the IP address of the computer. This address is in the form X.X.X.X, where each X is a number with between 1 and 3 digits.

Comments:

The socket must be open. If the specified socket has not been opened, the `peerAddress` function returns "not an open socket". If you have issued an open socket command to create the socket, you cannot use the `peerAddress` function until after the socket has been created and the command has completed.

The `connectionID` is needed only if more than one socket is connected to the same port of the same host. The `connectionID` is assigned by the `accept` or `open socket` command that created the socket.

penBack

property

Synonyms

Objects

button, field, graphic, global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

borderColor property

Summary

Has no effect and is included in Transcript for compatibility with imported SuperCard projects.

Syntax

set the penBack [of object] to colorNumber

Example Code

Comments

In SuperCard, the penBack property specifies the color used to draw borders. In Revolution, the border color is determined by the borderColor property.

The penBack property is always set to empty. A handler can set it to any value without causing a script error, but the actual value is not changed.

pencil

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

choose command, penColor property, tool function

Summary

Designates the paint tool used to draw freehand lines in an image.

Syntax

Example Code

```
choose pencil tool
```

Comments

Use the pencil keyword to draw lines with the penColor.

Comments:

When using the Pencil tool, the cursor is a pencil shape (over images) or an arrow (anywhere else). This tool is used to draw the penColor onto an image.

If you try to paint with the pencil on a card that has no images, an image the size of the card is created to hold the painting. If the current card already has one or more images, painting outside the image has no effect.

penColor

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

brushColor property, choose command, penPattern property, About colors and color references, Color Names Reference, Recipe for translating a color name to an RGB numeric triplet, Text menu > Color > Pen Color

Summary

Specifies the color used for drawing with the paint tools.

Syntax

set the penColor to {colorName | RGBColor}

Example Code

```
set the penColor to "blue"  
set the penColor to 255,255,255 -- white
```

Comments

Use the penColor property to change the color used with the pencil, line, and curve paint tools, and for the borders of shapes.

Value:

The penColor is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the penColor is "0,0,0" (black).

Comments:

If the penPattern has been set since the last time the penColor was set, the pattern is used instead of the color specified by penColor. In other words, the last-set property takes priority.

pendingMessages

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

cancel command, send command, How to determine which pending messages have been scheduled

Summary

Returns a list of messages that have been scheduled with the send command, but not yet delivered.

Syntax

the pendingMessages
pendingMessages()

Example Code

```
the pendingMessages  
repeat with x = 1 to the number of lines of the pendingMessages
```

Comments

Use the pendingMessages function to check whether a message has been sent yet, or to perform some action on each pending message.

Value:

The pendingMessages function returns a list of messages, one per line. Each line consists of four items, separated by commas:

- the message ID
- the time the message is scheduled for
- the message name
- the long ID property of the object that the message will be sent to

Comments:

The message ID is the same as the one placed in the result function when it was scheduled with the send command.

The time the message is scheduled for is in the same format as the long seconds form of the seconds function.

Once scheduled, a message cannot be changed. You can cancel a pending message with the cancel command, and re-send it with the send command.

penPattern

property

Synonyms

penPat

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

brushPattern property, choose command, foregroundPattern property, penColor property, tool property

Summary

Specifies the pattern used for painting with the paint tools.

Syntax

set the penPattern to {patternNumber | imageID | empty}

Example Code

```
set the penPattern to empty  
set the penPattern to 1013
```

Comments

Use the penPattern property to change the pattern used with the pencil, line, and curve paint tools, and for the borders of shapes.

Value:

The penPattern is a pattern specifier.

A patternNumber is a built-in pattern number between 1 and 164. (These patterns correspond to Revolution's built-in patterns 136 to 300.)

An imageID is the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the penPattern is empty.

Comments:

Pattern images can be color or black-and-white.

Cross-platform note: To be used as a pattern on Mac OS systems, an image must be 128x128 pixels or less, and both its height and width must be a power of 2. To be used on Windows and Unix systems,

height and width must be divisible by 8. To be used as a fully cross-platform pattern, both an image's dimensions should be one of 8, 16, 32, 64, or 128.

If the `penColor` has been set since the last time the `penPattern` was set, the color is used instead of the pattern specified by `penPattern`. In other words, the last-set property takes priority.

pi
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

acos function, asin function, atan function, atan2 function, constant command, cos function, sin function, tan function

Summary

Equal to the ratio of a circle's circumference to its diameter.

Syntax

Example Code

```
get tan((firstAngle * pi) + secondAngle)
```

Comments

Use the pi constant in trigonometric expressions where writing out "pi" makes the expression more readable and understandable.

Comments:

The pi constant is equal to 3.14159265358979323846.

pixels property

Synonyms

Objects image

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

backgroundPixel property, borderPixel property, bottomPixel property, focusPixel property, foregroundPixel property, hilitePixel property, shadowPixel property, topPixel property

Summary

The pixels property is not implemented and is reserved.

Syntax

Example Code

Comments

pixmapID

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

imagePixmapID property, import command, ink property, screenNoPixmaps property, windowID property

Summary

Specifies the ID of the data structure the operating system uses to hold the screen image of the current card.

Syntax

get the pixmapID of stack

Example Code

```
get colorAnimate(the pixmapID of this stack) -- an XFCN
```

Comments

Use the pixmapID property to pass to an external that needs to manipulate the window.

Value:

The pixmapID of a stack is an integer.

This property is read-only and cannot be set.

Comments:

The pixmap ID is provided by the operating system.

Some externals need to manipulate the contents of the image directly, and do so by referencing the ID. Such externals require you to pass the ID when you use the external.

place

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

backgroundBehavior property, copy command, create card command, groupNames property, insert script command, remove command, About groups and backgrounds, How to automatically include groups on a new card, How to include a group on a card, Object menu > Place Group

Summary

Adds a group of objects to a card.

Syntax

place group onto card

Example Code

```
place group "Navigation" onto card "Navy" -- group is on current card  
place background 1 onto this card -- group may be on any card
```

Comments

Use the place command to add an existing group to a card.

Parameters:

The group is any group in the current stack that is not contained (nested) in another group.

The card is any card in the current stack.

Comments:

When you use the create card command, if there are any groups on the current card whose backgroundBehavior property is set to true, they are automatically added to the new card, without needing to use the place command.

Important! The words "group" and "background" are not always interchangeable. A group reference with the word "group" refers to a group on the current card. To refer to a group that's not on the current card, you must use the "background" form.

Tip: To have a group automatically placed on newly created cards, set its `backgroundBehavior` property to `true`.

plain
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

bold keyword, italic keyword, textStyle property, unlock screen command, visual effect command, How to remove all styles from text, Text menu > Plain

Summary

Used with the visual effect command to designate a simple "cut" effect. Also used with the textStyle property to indicate plain text with no added styling.

Syntax

Example Code

```
hide field 3 with visual effect plain to gray  
set the textStyle of field "Alert" to plain
```

Comments

Use the plain keyword to create a visual effect that goes directly to the ending image, or to remove styles from an object or a chunk of text in a field.

Comments:

Setting the textStyle of an object or chunk to plain removes all styles (such as bold or italic) that have previously been applied to the object or chunk.

The plain visual effect cuts directly to the final image, as though a visual effect had not been used at all.

platform

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

charSet property, environment function, lookAndFeel property, machine function, processor function, systemVersion function, Supported Platforms Reference, Recipe for finding out whether OS X is running

Summary

Returns the name of the operating system Revolution is running on.

Syntax

the platform
platform()

Example Code

```
the platform  
if the platform is "MacOS" then set the activatePalettes to true
```

Comments

Use the platform function to change a stack's behavior or appearance depending on the operating system.

Value:

The platform function returns one of the following strings:

MacOS	any Mac OS, OS X, or Darwin system
Linux	Linux for Intel or PowerPC architecture
BSD	BSD UNIX (BSDI, FreeBSD)
HP-9000/700	HP-UX
SGI IRIS	Silicon Graphics IRIX
IBM RS/6000	AIX
Intel SCO	SCO Open DeskTop
Intel SVR4	Solaris for x86 architecture
SPARC	SPARC SunOS
SPARC Solaris	SPARC Solaris
Win32	Windows (any version post-3.1)

Comments:

The platform function is compiled into each version of the Revolution engine. This means that, for example, if you're developing on a Mac OS system and you build a standalone application for Linux, when you run the standalone application on a Linux system, the platform function returns "Linux".

Tip: To determine whether a Mac system is running Mac OS or OS X, use the systemVersion function along with the platform function, as in the following example:

```
set the itemDelimiter to "." -- versions are of the form "x.y.z"
if the platform is "MacOS" and item 1 of the systemVersion >= 10 then
  answer "This is an OS X system."
end if
```

Changes to Transcript:

Support for Alpha Digital UNIX was removed and support for OS X was added in version 1.1.

play

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

beep command, currentTime property, frameCount property, frameRate property, import command, movie function, playDestination property, playLoudness property, playRate property, playStopped message, prepare command, record sound command, revPlayAnimation command, sound function, start command, stop command, videoClipPlayer property, How to animate a sprite, How to beep, How to find out whether QuickTime is available, How to make the computer speak out loud, How to play a streaming QuickTime file, Why don't movies play?

Summary

Plays a movie or sound.

Syntax

play [filePath | type] [looping] [at point] [options xOptions]

play [stop | pause | resume | step {forward | back}] clip

Example Code

```
play "/usr/local/clips/music.aiff" -- a file
play videoClip "Movie" at 100,100 -- an imported video clip
play audioClip "Trust No One" looping
play pause videoClip "Sample"
```

Comments

Use the play command to control playing of a movie or sound.

Parameters:

The filePath is the location and name of the file you want to play. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

The type is "audioClip" or "videoClip".

The clip is a reference to an audio clip or video clip in an open stack.

The point specifies the center of the movie to be played, relative to the current stack window. If the point is not specified, the movie is positioned at the center of the current card. If a sound is being played, the point has no effect.

The xOptions are command line parameters passed to the "xanim" package on Unix systems. (On Mac OS and Windows systems, this parameter has no effect.)

Comments:

If you use the play...looping form, the sound or video plays continuously from start to beginning until you stop it.

You can stop playing of a movie or sound with the stop command or with the play stop form. If you specify a clip, only that clip is stopped. If you specify a type but not a clip, the last clip of that type is stopped. (On Unix systems, you must wait a second or two after starting a movie or sound before you can stop it, to give the external player time to start up.)

To pause a movie, use the play pause form. Continue playing with play resume. You can move one frame backward or forward with the play step form. If you use one of these commands with a clip that is not currently playing, the result function returns "videoClip is not playing". If the clip is a sound, these forms simply play the sound, ignoring the words "pause", "resume", or "step".

You can play multiple movies at once by starting each one of them with the play command.

Movies cannot be played while any tool other than the Browse tool is in use.

If you start playing an audio clip when another one is playing, the first audio clip is stopped, and a playStopped message is sent to the current card. You cannot play two sounds at the same time, nor can you queue a sound while another sound is playing.

On Unix systems, the "xanim" program must be located in a directory in the PATH environment variable. You can set the PATH from within Revolution by using the put command:

```
put newPath into $PATH
```

The play command does not work on some Unix systems that lack built-in support for sound. On these systems, when the play command executes, the result is set to "no sound support".

playDestination

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

play command, playLoudness property, recordInput property

Summary

The playDestination specifies where sound output is sent.

Syntax

set the playDestination to {internal | external}

Example Code

```
set the playDestination to external
```

Comments

Use the playDestination property to control whether sounds made with the play command on Unix systems are sent to the speakers or the audio jack.

Value:

The playDestination is "internal" or "external".

By default, the playDestination is set to "internal".

Comments:

If the playDestination property is set to "internal", sounds are played on the computer's internal speaker.

If the playDestination is "external", sounds are sent to the computer's audio port.

The setting of this property has no effect on Mac OS , OS X, or Windows systems.

player

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

button keyword, choose command, field keyword, graphic keyword, image keyword, scrollbar keyword, style property, templatePlayer keyword, tool function

Summary

Designates the Player tool, which is used to create new player objects for movies and sounds.

Syntax

Example Code

```
choose player tool
```

Comments

Use the player keyword to create players.

Comments:

When using the Player tool, the cursor is a crosshairs. This tool is used for creating players by clicking at one corner of the player and dragging to the opposite corner.

player object

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

templatePlayer keyword, About object types and object references, File menu > New Referenced Control > Quicktime-Supported File..., Object menu > New Control > Player

Summary

A control that displays a movie or sound file.

Syntax

Example Code

```
start player "San Antone"  
hide player (the selectedText of button "Current Movie")
```

Comments

Use the player object type to display a movie or sound from a separate file.

Comments:

Unlike an audio clip or video clip, a player does not contain the movie or sound data. Instead, you use the player's filename property to indicate the separate file that holds the movie or sound. This reduces the memory required by your stack, because the movie or sound data is only loaded into memory when it's being used, rather than being loaded into memory whenever the stack file is open. However, it also makes it possible for the movie or sound data to be misplaced during distribution, since the file is separate from your stack file.

A player is contained in a card, group, or background. Players cannot contain other objects.

The player object has a number of properties and messages associated with it. To see a list of messages that can be sent to a player as a result of user actions or internal Revolution events, open the "Transcript Language Dictionary" page of the main Documentation window, and choose "Player Messages" from the Show menu at the top. To see a list of all the properties a player can have, choose "Player Properties" from the Show menu.

playLoudness

property

Synonyms

Objects

audioClip, videoClip, player, global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

beep command, beepLoudness property, play command, playDestination property, recordLoudness property, showController property

Summary

Specifies the volume of sounds played by the play command.

Syntax

set the playLoudness [of {audioClip | player}] to percentage

Example Code

```
set the playLoudness to 100 -- maximum loudness
set the playLoudness of audioclip ID 1054 to 50
```

Comments

Use the playLoudness property to set the speaker volume.

Value:

The playLoudness is an integer between zero and 100.

By default, the playLoudness property is a number corresponding to the current system setting.

Comments:

Setting the playLoudness to zero turns off sounds.

If no audio clip, video clip, or player is specified, the setting of the playLoudness property applies to all sounds Revolution makes, including those made with the beep and play commands.

Users can manually set the playLoudness of a player using the slider on the left side of the player's controller bar, which is visible if the player's showController property is true.

playPaused

message

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentTime property, currentTimeChanged message, endTime property, paused property, play command, playStopped message

Summary

Sent to a player when the user pauses it.

Syntax

playPaused

Example Code

```
on playPaused -- start a "paused" animation
  set the repeatCount of image "Pause Animated Icon" to -1
end playPaused
```

Comments

Handle the playPaused message if you want to perform an update when a player is paused.

Comments:

A playPaused message is also sent when the user clicks within a player's rectangle and the player is already paused.

playRate

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

duration property, play command, playSelection property, playStopped message, start command, stop command, timeScale property

Summary

Specifies how fast a player plays a movie.

Syntax

set the playRate of player to rate

Example Code

```
set the playRate of player 17 to 0.5 -- half-speed
```

Comments

Use the playRate property to speed up or slow down playback of a movie or sound.

Value:

The playRate of a player is a real number.

By default, the playRate property of a newly created player is set to 1.

Comments:

The playRate is the ratio between the desired playback rate and the natural rate of the movie or sound. A playRate of 1 plays the movie or sound at normal speed; a playRate greater than one speeds up the playback; a playRate less than one slows down the playback.

A negative playRate plays the movie or sound backward.

If the playRate is zero, the movie or sound is paused.

playSelection

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

duration property, endTime property, play command, showSelection property, start command, startTime property, stop command

Summary

Specifies whether a player plays back its entire movie or sound, or only the selected portion.

Syntax

set the playSelection of player to {true | false}

Example Code

```
set the playSelection of player "Scan" to true
```

Comments

Use the playSelection property to play a selected portion of a movie or sound.

Value:

The playSelection of a player is true or false.

By default, the playSelection property of a newly created player is set to false.

Comments:

If the playSelection property is set to true, the play command only plays the section between the startTime and endTime.

If the playSelection is false, the play command plays the entire movie or sound, even if a smaller portion has been selected.

playStopped

message

Synonyms

Objects

player, videoClip, audioClip

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentTime property, currentTimeChanged message, endTime property, paused property, play command, playPaused message

Summary

Sent to a player when it stops playing.

Syntax

playStopped

Example Code

```
on playStopped -- hide the player when it finishes
  hide the target
  show button "Start Playing"
end playStopped
```

Comments

Handle the playStopped message if you want to perform a task when a movie or sound finishes playing.

Comments:

The playStopped message is sent when the movie or sound reaches its end, or when a play stop command executes. If the user pauses the movie or sound, the playPaused message is sent instead.

When an audio clip or video clip is playing, a temporary player is created for it. When the clip is finished, the playStopped message is sent to it.

Note: The playStopped message is sent when a card containing the player closes and when the player's filename property is changed, even if the movie or sound is not currently running. To prevent a playStopped handler from being executed inappropriately, set the lockMessages to true before changing the filename or switching cards:

```
lock messages -- prevent sending playStopped
set the filename of me to newFile
unlock messages
```


plus
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, cross constant, cursor property

Summary

Equivalent to the number 13.

Syntax

Example Code

```
set the cursor to plus
```

Comments

Use the plus constant to set the cursor to a plus sign, suitable for selecting spreadsheet cells.

Comments:

The following two statements are equivalent:

```
set the cursor to plus  
set the cursor to 12
```

However, the first is easier to read and understand in Transcript code.

Important! If you use the plus cursor or other standard cursors in your application, you must include it when you create your standalone. Make sure the "Cursors" option on the Inclusions tab in Step 3 of the Distribution Builder is checked.

point
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

is a operator, is not a operator

Summary

Used with the is a and is not a operators to specify whether a value is a point.

Syntax

Example Code

```
if it is not a point then exit mouseUp
```

Comments

Use the point keyword to find out whether a value can be used in expressions that require a point on the screen.

Comments:

If the value consists of two numbers separated by a comma or by whitespace, it is a point. The numbers may be non-integers.

If the value is an expression, it is evaluated and Revolution checks whether the final value is a point. For example, the value of the expression "22 + 3,23" is 25,23 (which is a point), so the following expression evaluates to true:

22 + 3,23 is a point

However, the expression

"22 + 3,23" is a point

evaluates to false, because the double quotes prevent the expression "22 + 3,23" from being evaluated.

pointer

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

browse keyword, choose command, tool function, Tools menu > Pointer Tool

Summary

Designates the Pointer tool, which is used to select objects.

Syntax

Example Code

```
choose pointer tool
```

Comments

Use the pointer keyword to edit a stack.

Comments:

When using the Pointer tool, the cursor is an arrow. Use this tool to select, resize, and move controls.

You cannot edit the objects in stacks whose `cantModify` property is true or stacks which are being displayed as a palette, modeless dialog, or modal dialog. Only stacks displayed as an editable window can be edited. (Use the `topLevel` command to display a stack in an editable window.)

pointerFocus

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

defaultStack property, focusedObject function, topStack function

Summary

Specifies whether typed text is sent to the window under the mouse pointer or to the frontmost window.

Syntax

set the pointerFocus to {true | false}

Example Code

```
set the pointerFocus to true
```

Comments

Use the pointerFocus property to determine how the active window is determined.

Value:

The pointerFocus is true or false.

By default, the pointerFocus property is set to true on Mac OS, OS X, and Windows systems, and to false on Unix systems that ship with Motif.

Comments:

On Unix systems, if the pointerFocus property is false, clicking inside a window makes it the active window. This is the recommended setup.

If the pointerFocus property is true, moving the mouse pointer into a window makes it the active window. Set the pointerFocus to true to work around problems in some Unix window managers (specifically, "olwm" and "fvwm") that prevent active-focus applications such as Revolution from operating correctly.

If the application is started from a Unix command line, this property can be set to true on startup by using the -pointerfocus option.

Cross-platform note: On Mac OS, OS X, and Windows systems, setting the `pointerFocus` property to `false` changes window activation in minor ways which are not particularly useful, and has no other effect.

points

property

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

dashes property, endArrow property, markerDrawn property, markerPoints property, relativePoints property, startArrow property

Summary

Specifies where the vertexes of a graphic are located.

Syntax

set the points of graphic to vertexList

Example Code

```
set the points of the selectedObject to myVertexes
```

Comments

Use the points property to find the shape of a line, curve, or irregular polygon graphic, or to reshape the graphic.

Value:

The points of a graphic is a list of points (two integers separated by a comma), one per line.

Comments:

Each point consists of the horizontal distance in pixels from the left edge of the stack window to the vertex, a comma, and the vertical distance in pixels from the top edge of the stack window to the vertex. (The relativePoints property is the same list, but measured from the top left of the graphic rather than the top left of the stack window.)

The first line in the list is the location of the graphic's starting point. A line is drawn from the starting point to the next vertex, which is the next line in the list. If two successive lines are identical, a dot is drawn at the specified point.

A blank line in the points indicates that the previous and next vertexes are not connected by a line—that is, the line, curve, or polygon is broken into two (or more) pieces. If the last line of the points of a

polygon is blank, the polygon is not closed. A closed polygon's start point (the first line of its points property) is the same as its end point (the last line of its points property).

Tip: When setting the points property, you can separate the individual points with a comma instead of a return. The points property is always reported with the points on separate lines, however.

If the style property of the graphic is not "line", "polygon" or "curve", the setting of its points property has no effect.

Note: The rectangle of a graphic is drawn around all its points without touching them. (Usually, this makes no difference, but in some circumstances where you need to place a graphic's vertex precisely with respect to another object's rectangle, you may need to take this into account.)

polygon

keyword

Synonyms
poly

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

brushColor property, brushPattern property, choose command, lineSize property, penColor property, points property, regular keyword, tool function, Object menu > New Control > Polygon Graphic

Summary

Designates the paint tool used to draw irregular polygon shapes. It also specifies, through the style property, that a graphic is an irregular polygon shape.

Syntax

Example Code

```
choose polygon tool  
set the style of the templateGraphic to polygon
```

Comments

Use the polygon keyword to paint an irregular polygon with the penColor or to change a graphic to an irregular polygon shape.

Comments:

When using the Polygon tool, the cursor is a crosshairs shape (over images) or an arrow (anywhere else). This tool is used to draw an irregular polygon in the penColor, filled with the brushColor (or brushPattern).

If you try to paint with the Polygon tool on a card that has no images, an image the size of the card is created to hold the painting. If the current card already has one or more images, painting outside the image has no effect.

Setting the style of a graphic to "polygon" makes the graphic into an irregular polygon. Irregular polygon graphics, unlike painted polygons, can be changed and reshaped: use the points properties to change the shape.

polySides

property

Synonyms

Objects

graphic, global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

angle property, choose command, regular keyword, style property, tool property

Summary

Specifies how many sides a regular polygon has.

Syntax

set the polySides [of graphic] to numberOfSides

Example Code

```
set the polySides to 3 -- to make triangles  
set the polySides of graphic "Hex" to 6 -- hexagon
```

Comments

Use the polySides property to change the appearance of polygons.

Value:

The polySides of a graphic is an integer greater than 2.

By default, the polySides of a newly created graphic is 4.

Comments:

The polySides property affects only graphics whose style property is set to regular.

The global setting of the polySides property controls the appearance of polygons drawn with the paint tools. Once a paint polygon is drawn, its appearance can no longer be changed by changing the global polySides property.

pop

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

go command, ID property, lockRecent property, push command

Summary

Retrieves a card from the list created with the push command, and either goes to that card or places its ID in a container.

Syntax

pop card [{into | before | after} container]

Example Code

```
pop card -- go to the popped card
pop card after reverseListOfCards
```

Comments

Use the pop command to return to a card whose location you have previously stored.

Parameters:

The container is a field, button, URL, or variable, or the message box.

Comments:

If you do not specify a container, the pop command goes to the last card you placed on the list with the push command, and removes that card from the list.

If you specify a container, the long ID of the card is placed in the container and removed from the list. The current card does not change.

If the list is empty when you use the pop command, Revolution goes to the Home card. If you are running in the Revolution development environment, this is the first card of the "license.rev" stack, which is the splash screen that appears when you start up Revolution. (Click this card to close the stack.) If you are running in a standalone application, the Home card is the first card of the standalone's main stack.

popup

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clickLoc function, menuMode property, mouseLoc function, option command, pulldown command, style property

Summary

Displays a popup menu whose menu items are either lines in a button or buttons in a stack.

Syntax

popup {stack | button} [at location]

Example Code

```
popup button "Contextual Menu"  
popup stack "Button Properties"  
popup stack (field "Which")
```

Comments

Use the popup command to display a contextual menu.

Parameters:

The stack is any stack reference. The stack's first card contains a button for each menu item in the popup menu.

The button is a reference to a button on the current card.

The location is an expression that evaluates to a pointótwo integers separated by a comma.

Comments:

Use the popup command in a mouseDown handler to display the menu.

The popup menu appears with its top left corner at the location. If no location is specified, the menu's top left corner is at the mouse location. While the popup menu is displayed, the handler pauses.

You can use the mouse button parameter of the mouseDown message to specify that the menu should appear only when the user right-clicks (on Unix and Windows systems) or Control-clicks (on Mac OS systems):

```
on mouseDown theButton
  -- pop up the menu on right-click or control-click
  if theButton is 3 then popup stack "Lookup Options"
  else pass mouseDown
end mouseDown
```

Choosing a menu item from the popup menu sends a menuPick message to the button (for a button menu) or a mouseUp message to the clicked control in the stack (for a stack menu). The button or stack menu handles the menu choice.

If you use a button to hold the contents of the menu, the button's style property must be set to "menu" and its menuMode must be set to "popup". The button's visible property may be set to either true or false.

Important! The menuMouseButton property of a button used with the popup command must be set to zero. Setting it to any other value may result in unexpected behavior when the menu is used. To control which mouse buttons may be used to access the menu, use the mouse button parameter of the mouseDown message, as described above.

Changes to Transcript:

The option to pop up a button menu was introduced in version 1.1. In previous versions, only stack menus could be used with the popup command.

The location parameter was introduced in version 2.0. In previous versions, the popup menu always appeared with its top left corner at the mouse location.

popup

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cascade keyword, comboBox keyword, menuMode property, option keyword, popup command, pulldown keyword, style property, tabbed keyword, Object menu > New Control > Popup Menu

Summary

Specifies one of the menu types that can be used with the menuMode property.

Syntax

Example Code

```
set the menuMode of button "Hot Text Options" to popup
```

Comments

Use the popup keyword to create a popup menu that appears under the mouse pointer.

Comments:

Popup menus are normally used as contextual menus.

post

command

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

httpHeaders property, httpProxy property, libURLSetCustomHTTPHeaders command, open socket command, read from socket command, URLEncode function, URLStatus function, write to socket command, How to access the Internet from behind a firewall

Summary

Sends data to a web server using the POST action of HTTP.

Syntax

post data to URL destinationURL

Example Code

```
post myData to URL "http://www.example.net/indications.cgi"  
post field "Return Values" to URL field "Current Page"
```

Comments

Use the post command to submit data to a web server.

Parameters:

The data is any string.

The destinationURL is the URL where the data is to be sent.

Comments:

Data you send should be encoded using the URLEncode function.

The value the web server returns is placed in the it variable. If an error occurs, the result function is set to the error message.

The HTTP header sent with the POST request can be changed using either the HTTPHeaders property or the libURLSetCustomHTTPHeaders command. By default, the "Content-Type" header line is set to "application/x-www-form-urlencoded".

Note: Sending data with the post command is a blocking operation: that is, the handler pauses until Revolution is finished sending the data. Since contacting a server may take some time due to network lag, URL operations may take long enough to be noticeable to the user.

Important! If a blocking operation involving a URL (using the put command to upload a URL, the post command, the delete URL command, or a statement that gets an ftp or http URL) is going on, no other blocking URL operation can start until the previous one is finished. If you attempt to use a URL in an expression, or put data into a URL, while another blocking URL operation is in progress, the result is set to "Error Previous request not completed".

To send a username and password with the post command, use the standard syntax for including this information in a URL. For example, to access <http://www.example.com/> with the username "me" and the password "pass", use the following statement:

```
post someData to URL "http://me:pass@example.com"
```

Important! If your user name or password contains any of the characters ":", "@", "/", ".", or "|", use the URLEncode function to safely encode the user name or password before putting them into the URL. The following example constructs a URL for a user whose password contains the "@" character:

```
put "name" into userName  
put "jdoe@example.com" into userPassword  
put "http://" & userName & ":" & URLEncode(userPassword)
```

```
& "@www.example.net/index.html" into fileURLToGet  
get URL fileURLToGet
```

Important! The post command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the post command is not yet available. This may affect attempts to use this command in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the post command can be used in any handler.

postScript

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

boundingBox property, currentPage property, pageCount property, prolog property, retainPostScript property

Summary

Specifies the actual PostScript code associated with an EPS object.

Syntax

set the postScript of EPSObject to code

Example Code

```
set the postScript of EPS "To Print" to field "Result"
```

Comments

Use the postScript property to examine or change the PostScript code of an EPS object.

Value:

The postScript of an EPS object is a string.

By default, the postScript property of newly created EPS objects is set to empty.

Comments:

This property is supported only on Unix systems with Display PostScript installed.

powerKeys

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

centered property, filled property, keyDown message, navigationArrows property, tool property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the powerKeys to {true | false}

Example Code

Comments

In HyperCard, the powerKeys property determines whether certain keyboard shortcuts can be used with the paint tools. In Revolution, these shortcuts are not used.

The powerKeys property is set to true by default. A handler can change this property without causing a script error, but the setting has no effect.

preOpenBackground

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

closeBackground message, openBackground message, preOpenCard message, preOpenStack message

Summary

Sent to the current card when you go from a card that does not have a group to a card that does.

Syntax

preOpenBackground backgroundID

Example Code

```
on preOpenBackground theGroup -- move the group's position
  if the short name of background ID theGroup is "Navigation"
    then set the bottom of group "Navigation" to the height of this card
  end preOpenBackground
```

Comments

Handle the preOpenBackground message to update a background's appearance before the background appears on screen.

Parameters:

The backgroundID is the short ID property of the background being opened.

Comments:

The actual navigation is not triggered by the preOpenBackground message, so trapping the message and not allowing it to pass does not prevent the card with the group from opening.

If there is more than one group on the card that was not on the previously-visited card, a preOpenBackground message is sent to each group.

preOpenCard

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeCard message, openCard message, preOpenBackground message, preOpenStack message

Summary

Sent to a card when you go to the card.

Syntax

preOpenCard

Example Code

```
on preOpenCard -- highlight the "go to this card" button in a set
  set the hilitedButton of group "Cards" to the number of this card
  pass preOpenCard
end preOpenCard
```

Comments

Handle the preOpenCard message to update a card's appearance before the card appears on screen.

Comments:

The preOpenCard message is sent before the openCard message. Unlike openCard, preOpenCard handlers are executed before the card appears. Because of this, the preOpenCard handler is a good place to put code that adjusts the size, position, and appearance of objects; the changes are made before the card appears.

The actual navigation is not triggered by the preOpenCard message, so trapping the message and not allowing it to pass does not prevent the card from opening.

preOpenStack

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

closeStack message, openStack message, preOpenBackground message, preOpenCard message, How to respond when a stack opens, Why does my stack open slowly?

Summary

Sent to the destination card when you open a stack.

Syntax

preOpenStack

Example Code

```
on preOpenStack -- center the stack on the main screen
  set the loc of this stack to the screenLoc
  pass preOpenStack
end preOpenStack
```

Comments

Handle the preOpenStack message to update a card's appearance before the card appears on screen.

Comments:

The preOpenStack message is sent before the openStack message. Unlike openStack, preOpenStack handlers are executed before the stack window appears. Because of this, the preOpenStack handler is a good place to put code that adjusts the size, position, and appearance of objects; the changes are made before the stack appears.

The actual navigation is not triggered by the preOpenStack message, so trapping the message and not allowing it to pass does not prevent the stack from opening.

prepare command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

frameRate property, movie function, play command

Summary

Preloads a movie into memory.

Syntax

prepare videoClip filePath [looping] [at startPoint]

Example Code

```
prepare videoClip "/Disk/Folder/Movie" at 100,100  
prepare videoClip ID 17 looping
```

Comments

Use the prepare command to speed up execution of the play command.

Parameters:

The filePath is the location and name of the file you want to play. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

The point specifies the center of the movie to be played, relative to the current stack window. If the point is not specified, the movie is positioned at the center of the card.

Comments:

Use the prepare command to preload the movie at some time before it is needed. Then use the play command with identical parameters to actually play the movie. Since the movie is already loaded into memory, it starts up instantly when the play command executes.

The parameters of the prepare command must be identical to those of the corresponding play command.

previous
keyword

Synonyms
prev

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
first keyword, go command, last keyword, next keyword

Summary
Indicates the card that comes before the current card.

Syntax

Example Code
go previous card
get the short name of previous card

Comments
Use the previous keyword to move backward in the stack or to reference the card before the current card.

Comments:
If you are already on the first card, the previous card is the last card in the stack.

print

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

answer printer command, close printing command, formatForPrinting property, open printing command, printCommand property, printFontTable property, printGutters property, printMargins property, printPaperSize property, printRotated property, printRowsFirst property, printScale property, revPrintField command, revPrintReport command, revPrintText command, How to print in landscape mode, Recipe for fitting a printout to the page, Recipe for printing the fields on a card, File menu > Print Card..., File menu > Print Field...

Summary

Prints one or more cards.

Syntax

```
print card [from topLeft to rightBottom] [into pageRect]
print {stack|card|{marked|number|all} cards} [into pageRect]
print {stack|card|{marked|number|all} cards} [from topLeft to rightBottom]
print break
```

Example Code

```
print card "Output"
print stack "Heavenly Days" into 100,100,450,410
print this card from 0,0 to the mouseLoc
print marked cards
print 20 cards into the rect of field "Preview"
```

Comments

Use the print command to print out a card, a set of cards, or all the cards of a stack.

Parameters:

The stack is any open stack. If you specify a stack, all the cards of that stack are printed. (You can print a stack even if its window isn't visible.)

The card is any card reference. If you specify a card, that single card is printed.

The number is the number of cards to print, starting with the current card.

The `pageRect` is the rectangle into which the card is printed, and consists of four integers separated by commas: the top, left, bottom, and right edges of the printed card, in points. (There are 72 points to an inch.) The card is scaled to fit the specified `pageRect`. If you don't specify a `pageRect`, the card's size depends on the `printScale` property.

The `topLeft` and `rightBottom` are points specifying the portion of the card to be printed. Each point consists of two integers separated by a comma:

- the horizontal distance in pixels from the left edge of the card window to the point

- the vertical distance from the top edge to the point

If you don't specify a `topLeft` and `rightBottom`, the entire visible portion of the card is printed.

Comments:

The print command prints only the area of a card that is visible in the stack window. To print an area of a card whose objects extend beyond the window boundaries, either set the stack's rectangle so it's large enough to hold all the objects to print, or use the `print...from` form of the print command.

To print multiple cards selected one at a time, start with the open printing command, issue any print commands you want, then use the close printing command to print all the cards you specified in the print commands as a single batch.

If you specify a form that includes more than one card (such as `print stack`), the cards may be printed more than one to a page, depending on the size of the cards and on whether you specify a `pageRect`.

The `print marked cards` form prints all the cards in the current stack whose `mark` property is set to true.

The `print all cards` form is equivalent to `print this stack`.

The `print break` form forces a page break.

Note: If a card is larger than a full page, the print command prints only the first page of the card, starting at the top left corner. To print the entire card, use the `print...into pageRect` form to scale the card to the page.

Revolution visits each card as it's printed, returning to the original card when the printing is done, but it doesn't send any system messages such as `openCard` while moving from card to card during printing. If you don't want the user to see these card changes, set the `lockScreen` property to true before you print.

Cross-platform note: On Mac OS and Windows systems, the print command uses the current printer. On Unix systems, the print command creates a PostScript file and runs the program specified in the `printCommand` property, with the file as input.

printCardBorders

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

print command, printMargins property, showBorder property

Summary

Specifies whether the printed image of a card is outlined.

Syntax

set the printCardBorders to {true | false}

Example Code

```
set the printCardBorders to false
```

Comments

Use the printCardBorders property to place a border around each card when printed.

Value:

The printCardBorders is true or false.

By default, the printCardBorders property is set to false.

Comments:

If the card's showBorder property is false, the border is not drawn and the setting of this property has no effect.

printColors

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

foregroundColor property, print command, printCardBorders property

Summary

Specifies whether objects are printed in color or black and white.

Syntax

set the printColors to {true | false}

Example Code

```
set the printColors to false
```

Comments

Use the printColors property to adjust printing depending on the type of printer and what's being printed.

Value:

The printColors is true or false.

By default, the printColors property is set to true.

Comments:

Set the printColors property to true if you're using a color printer and want to print colors on the page.

If you're using a black-and-white printer, the print driver may dither colored text (print it in a black-and-white pattern to approximate the shade of color). In this case, setting the printColors property to false may make the text more readable when printed.

printCommand

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

print command, printColors property, printScale property

Summary

Specifies the command line used for printing a PostScript file.

Syntax

set the printCommand to commandLine

Example Code

```
set the printCommand to "/usr/bin/printme %s"  
set the printCommand to "mv %s output.ps" -- puts output in a file
```

Comments

Use the printCommand property to control which program is used to print on Unix systems.

Value:

The printCommand is a string.

By default, the printCommand property is set to "lp %s". ("lp" is the standard line-printer command.)

Comments:

The value of the printCommand must be a valid Unix command line. The string "%s" is a placeholder for the print file name Revolution sends to the printing command.

The setting of this property has no effect on Mac OS or Windows systems.

printer:
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

close file command, COM1: keyword, LPT1: keyword, modem: keyword, open file command, read from file command, serialControlString property, write to file command

Summary

Used with the open file, read from file, write to file, and close file commands to specify the printer port on Mac OS systems.

Syntax

Example Code

```
open file "printer:"
```

Comments

Use the printer: keyword to communicate through the printer serial port.

Comments:

To set the port parameters (such as baud rate, handshaking, and parity), set the serialControlString property before opening the port with the open file command.

To read data from the printer port, use the read from file command, specifying the keyword printer: as the file to read from.

To write data to the printer port, use the write to file command.

To use the modem port on Mac OS systems, use the modem: keyword. (Revolution does not support additional serial ports.) To use serial ports on Windows systems, use the COM1: through COM9: keywords.

printFontTable

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

print command, printColors property, textFont property

Summary

Specifies the PostScript fonts that are substituted for screen fonts when you print on a PostScript printer.

Syntax

set the printFontTable to fontCorrespondencesList

Example Code

```
set the printFontTable to bookmanFonts & return & arialFonts
```

Comments

Use the printFontTable property to specify substitution mappings for nonstandard fonts you use, or to change the standard font substitutions for better results.

Value:

The printFontTable is a list of font names and their print equivalents, one per line.

Comments:

Each line of the printFontTable corresponds to one screen font and contains the following items, separated by commas:

- ī The name of an X11 screen font
- ī The name of the PostScript font to use in its place for printing
- ī The name of the normal style for that font
- ī The name of the bold style for that font
- ī The name of the italic style for that font
- ī The name of the bold italic style for that font

By default, the printFontTable property uses a mapping between the standard MIT X11R4 fonts and the PS35 font set.

The setting of this property has no effect on Mac OS and Windows systems.

printGutters

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

print command, printMargins property, printPaperSize property, printRotated property, printRowsFirst property

Summary

Specifies how much blank space is used to separate printed cards on the page.

Syntax

set the printGutters to columnSpace,rowSpace

Example Code

```
set the printGutters to 9,27
```

Comments

Use the printGutters property to control the amount of blank space between cards, when multiple cards are printed with the open printing command.

Value:

The printGutters consists of two non-negative integers, separated by commas.

By default, the printGutters is set to 36,36 (one-half inch between cards, vertically and horizontally).

Comments:

The first item of the printGutters is the height in points of the blank strip between the right edge of one column of cards and the left edge of the next column. (There are 72 points to an inch.)

The second item is the width in points of the blank strip between the bottom edge of one row of cards and the top edge of the next row.

printMargins

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

print command, printGutters property, printPaperSize property, printRotated property, printScale property

Summary

Specifies the width of the page margins when printing cards.

Syntax

set the printMargins to left,top,right,bottom

Example Code

```
set the printmargins to 72,144,72,144 -- 2" on top and bottom
```

Comments

Use the printMargins property to control how much blank space is left at each edge of the page when printing.

Value:

The printMargins consists of four non-negative integers, separated by commas.

By default, the printMargins is set to 72,72,72,72 (a one-inch margin on each side).

Comments:

The left is the width in points of blank space between the left edge of the page and the leftmost edge of the printed cards. (There are 72 points to an inch.)

The top is the height in points of blank space between the top edge of the page and the topmost edge of the printed cards.

The right is the width in points of blank space between the right edge of the page and the rightmost edge of the printed cards.

The bottom is the height in points of blank space between the bottom edge of the page and the bottommost edge of the printed cards.

printPaperSize

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

print command, printMargins property, printRotated property, printScale property

Summary

Specifies the dimensions of the paper used for printing.

Syntax

set the printPaperSize to paperWidth,paperHeight

Example Code

```
set the printPaperSize to 612,792 -- 8.5x11 inches, US Letter size
```

Comments

Use the printPaperSize property to control the size of the printout.

Value:

The printPaperSize consists of two non-negative integers, separated by a comma.

By default, the printPaperSize is the pixel size set by the print driver. On Mac OS and OS X systems, this is set in the Page Setup dialog box; on Windows systems, it is set in the printing options dialog box.

Comments:

When printing multiple cards using the open printing command, Revolution uses the printPaperSize property to determine how many cards fit on a page.

The paperWidth and paperHeight are specified in points. (There are 72 points in an inch.)

Important! If you have used the answer printer command during the current session, the paper size is set to the size selected in the dialog box, and it is not possible to change the printPaperSize property's value for the remainder of the current session. To use a different paper size after using the answer printer command, you must first quit and restart the application.

printRotated

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer printer command, print command, printRowsFirst property, printScale property

Summary

Specifies whether Revolution prints in landscape mode or portrait mode.

Syntax

set the printRotated to {true | false}

Example Code

```
set the printRotated to true
```

Comments

Use the printRotated property to print wide pages.

Value:

The printRotated is true or false.

By default, the printRotated property is set to false.

Comments:

If the printRotated property is set to false, printing is done in portrait modeóthe short sides of the paper are the top and bottom of the printed page.

If the printRotated is true, the direction of the printed output is rotated 90 degrees, so the long sides of the paper are the top and bottom.

Tip: On Mac OS systems, to set the orientation of the printout, use the answer printer command to display the Page Setup dialog box.

printRowsFirst

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

open printing command, print command, printMargins property, printRotated property, printScale property

Summary

Specifies whether cards are printed down first or across first.

Syntax

set the printRowsFirst to {true | false}

Example Code

```
set the printRowsFirst to false
```

Comments

Use the printRowsFirst property to determine the layout of cards on the page when printing multiple cards.

Value:

The printRowsFirst is true or false.

By default, the printRowsFirst property is set to true.

Comments:

When printing multiple cards using the open printing command, the application uses the printRowsFirst property to determine how the cards are arranged on the page. If the printRowsFirst is set to true, the cards are laid out across from left to right, then from top to bottom. If the printRowsFirst is false, the cards are laid out down from top to bottom, then from left to right.

By default, the printRowsFirst property is set to true.

printScale

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

print command, printMargins property, printPaperSize property, printRotated property, Recipe for fitting a printout to the page

Summary

Specifies how much cards are shrunk or expanded when printing.

Syntax

set the printScale to ratio

Example Code

```
set the printScale to 2 -- double-size cards
set the printScale to .25 -- quarter-size cards
```

Comments

Use the printScale property to control the size of printed cards.

Value:

The printScale is a positive number.

By default, the printScale property is set to 1.

Comments:

The printScale specifies the ratio between the number of pixels in the card and the number of points on the printed page. A ratio of 1 means the printed and screen card should be the same size, if the screen's resolution is 72 pixels per inch.

For example, if a card is 360 pixels wide and you want its printed image to be 4 inches (288 points) across, set the printScale to 0.8 (288 pixels divided by 360 points).

printTextAlign

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

print command, textAlign property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the printTextAlign to {left | center | right}

Example Code

Comments

In HyperCard, the printTextAlign property specifies the alignment of printed text. In Revolution, only cards can be printed, and the alignment of text on the card is determined by the card layout and the properties of its objects.

By default, the value of printTextAlign is center. A handler can set the property, but the new setting has no effect.

printTextFont

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

print command, scriptTextFont property, textFont property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the printTextFont to fontName

Example Code

Comments

In HyperCard, the printTextFont property specifies the font face of printed text. In Revolution, only cards can be printed, and the font of text on the card is determined by the textFont property of the objects containing text.

A handler can set the property, but the new setting has no effect.

printTextHeight

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

print command, textHeight property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the printTextHeight to lineHeight

Example Code

Comments

In HyperCard, the printTextHeight property specifies the vertical spacing of printed text. In Revolution, only cards can be printed, and the vertical spacing of text on the card is determined by the card layout and the properties of its objects.

By default, the value of printTextHeight is 18. A handler can set the property, but the new setting has no effect.

printTextSize

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

print command, scriptTextSize property, textSize property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the printTextSize to fontSize

Example Code

Comments

In HyperCard, the printTextSize property specifies the font size of printed text. In Revolution, only cards can be printed, and the size of text on the card is determined by the textSize property of the objects that display text..

By default, the value of printTextSize is 14. A handler can set the property, but the new setting has no effect.

printTextStyle

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

print command, textStyle property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the printTextStyle to {plain | stylesList}

Example Code

Comments

In HyperCard, the printTextStyle property specifies the styles of printed text. In Revolution, only cards can be printed, and the style of text on the card is determined by the textStyle property of objects that display text..

By default, the value of printTextStyle is plain. A handler can set the property, but the new setting has no effect.

privateColors

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

backgroundPixel property, colorMap property, foregroundPixel property, lockColorMap property, screenDepth function, Why does an image have the wrong colors?

Summary

Specifies whether Revolution uses its own color table or the system color table on Unix systems.

Syntax

set the privateColors to true

Example Code

```
set the privateColors to true
```

Comments

Use the privateColors property to improve display on Unix systems when the bit depth of the screen is 8 bits (256 colors) or less.

Value:

The privateColors is true or false.

By default, the privateColors property is set to false.

Comments:

Set the privateColors property to true for a stack that uses colors that aren't in the default color table.

This has the advantage of letting the stack display more colors than normally possible on an 8-bit display. The disadvantage is that if the privateColors is true, the colors of other applications' windows may be distorted while Revolution is the foreground application.

When the privateColors is set to false, the engine uses the system color table. When it is set to true, the engine uses its own custom color table.

This property has no effect unless the `screenType` property has a value of "PseudoColor"óthat is, each pixel on the screen is one of a color table of colors (usually 256 colors), and the colors in that color table can be changed by the engine.

Important! Once the `privateColors` property is set to true, it cannot be set back to false. To change it back to true, you must quit and restart the application.

The setting of this property has no effect on Mac OS or Windows systems.

processID

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

open process command, openProcesses function, openProcessIDs function, signal message

Summary

Returns the process ID of Revolution (or a standalone application).

Syntax

the processID

processID()

Example Code

```
the processID
```

```
write the processID & linefeed to process someOtherProcess
```

Comments

Use the processID function to communicate with another application.

Value:

The processID function returns a positive integer.

Comments:

On Unix systems, the processID uniquely identifies the process belonging to the running application. Other processes can use the processID to address Revolution.

The processID function does not return a meaningful value on Mac OS or Windows systems.

processor function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Not supported on UNIX systems

See Also:

machine function, platform function, Supported Platforms Reference

Summary

Returns a string describing the computer's CPU chip.

Syntax

the processor
processor()

Example Code

```
the processor  
if the processor is "Motorola PowerPC" then displayFastGraphics
```

Comments

Use the processor function to determine which microprocessor the computer has.

Value:

The processor function returns one of the following strings:

- Motorola PowerPC on a Mac OS system with a PowerPC processor
- Motorola MC68000 on a Mac OS system with a 68000-family chip (older Macs)
- unknown on Unix systems
- x86 on Windows systems

Comments:

The values returned by this function may change in future releases.

prolog

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

boundingBox property, postScript property

Summary

Specifies the PostScript prolog code of an EPS object.

Syntax

set the prolog of EPSObject to PostScriptString

Example Code

```
set the prolog of the templateEPS to field "Rotate"
```

Comments

Use the prolog property to prepend PostScript code to an EPS object.

Value:

The prolog of an EPS object is a string.

By default, the prolog property of newly created EPS objects is set to empty.

Comments:

You can use the prolog to transform the EPS without having to change its postScript property (the actual PostScript code used to render the image).

This property is supported only on Unix systems with Display PostScript installed.

properties

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

customProperties property, keys function, propertyNames function, set command, About properties and property profiles, Object menu > Object Inspector, Object menu > Card Inspector, Object menu > Stack Inspector

Summary

Specifies some of an object's properties and their current values.

Syntax

set the properties of object to propertiesArray

Example Code

```
put the properties of button 1 into myArray  
set the properties of last player to the properties of player "Example"
```

Comments

Use the properties property to set an object's built-in properties, or to copy properties from one object to another.

Value:

The properties of an object is an array containing that object's significant built-in properties.

Comments:

Not every property is included in the properties property. The following types of properties are excluded:

- Read-only properties
- The script property and custom properties
- Synonyms: Only one synonym for each property is included.

ï Duplicates: Properties that are functionally duplicated by other properties are not included. For example, the rectangle property is included, but not the height, width, top, bottom, left, right, or location properties, because you can derive all of them from the object's rectangle.

ï Properties other than object properties: Global properties, local properties, properties of a character or chunk in a field, and adjectives such as short that are implemented internally as properties are all excluded.

ï Others: Some other properties are excluded.

The key of each element in the array is the property name. Use the following statements to obtain a list of the properties for a particular object type:

```
put the properties of button "Example" into arrayVariable
put the keys of arrayVariable into propertiesList
```

The value of each element in the array is the value of that property for the object. For example, use this statement (after the above example) to get the object's ID property:

```
put the properties of button "Example" into arrayVariable
put arrayVariable[ID] into myID
```

Note: You can't use array notation with an expression, only with a variable, so you must put the properties of the object into a variable before you can access the individual elements of the array. If you want a list of properties and their values, use the combine command to create a list:

```
get the properties of field "Whatever"
combine it using return and colon -- convert from array to text list
put it into myFieldPropertiesList
```

If you set the properties of an object to an array that contains only some of the properties, any properties that aren't in that array are not changed. This means that you can select which properties to set with the properties property .

For example, suppose you want to set the properties of a field to the properties of another field, except that you want to leave the first field's name unchanged:

```
put the properties of field "My Field" into myArray
delete variable myArray[name] -- delete this element from the array
set the properties of field "New Field" to myArray
-- leaves name property unchanged
```

propertyNames

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

commandNames function, constantNames function, customProperties property, functionNames function, getProp control structure, properties property, setProp control structure, variableNames function, About properties and property profiles

Summary

Returns a list of all built-in properties in Transcript.

Syntax

the propertyNames

propertyNames()

Example Code

```
the propertyNames
if it is among the lines of the propertyNames then beep
```

Comments

Use the propertyNames function to check whether a particular property already exists in Transcript, to avoid using a reserved word for your own custom properties.

Value:

The propertyNames function returns a list of property names, one per line.

Comments:

The propertyNames function returns all the properties that can be used in Transcript, including synonyms.

proportionalThumbs

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

foregroundColor property, lookAndFeel property, platform function, thumbPosition property, thumbSize property

Summary

Specifies whether the draggable thumbs in scrollbars are a fixed size, or a size proportional to the amount of content shown.

Syntax

set the proportionalThumbs to {true | false}

Example Code

```
set the proportionalThumbs to true
```

Comments

Use the proportionalThumbs property to control the appearance of scrollbars.

Value:

The proportionalThumbs is true or false.

By default, the proportionalThumbs property is set to false on Mac OS systems, true on Unix and Windows systems.

Comments:

If the proportionalThumbs is set to true, the scrollbar thumb is sized proportionately to the visible content of its field, group, or stack. If almost all the content is currently visible, the thumb is large and fills most of the scrollbar; if only a small percentage of content is visible, the thumb is small.

pulldown

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

menuMode property, option command, popup command, style property

Summary

Displays a pulldown menu whose whose menu items are buttons in a stack.

Syntax

pulldown stack

Example Code

```
pulldown stack "See Also"
```

Comments

Use the pulldown command to display a stack menu as a pulldown menu in situations where it is not possible to attach the menu to a button (for example, if the contents of the menu are not known in advance).

Parameters:

The stack is any stack reference. The stack's first card contains a button for each menu item in the pulldown menu.

Comments:

Use the pulldown command in a mouseDown handler to display the menu:

```
on mouseDown  
  pulldown stack "Lookup Options"  
end mouseDown
```

While the menu is displayed, the handler pauses.

The menu appears aligned with the left edge of the control containing the handler. The menu appears below the control, if there is room on the screen. Choosing a menu item from the menu sends a mouseUp message to the stack.

Note: On Mac OS and OS X systems, pulldown menus in a window are drawn by the standard operating system routines if the button's `showBorder` property is set to `true` and its `borderWidth` is not zero. Pulldown menus in the menu bar are always drawn by the operating system.

pulldown

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cascade keyword, comboBox keyword, menuMode property, option keyword, popup keyword, pulldown command, style property, tabbed keyword, Object menu > New Control > Pulldown Menu

Summary

Specifies one of the menu types that can be used with the menuMode property.

Syntax

Example Code

```
set the menuMode of button "File" to pulldown
```

Comments

Use the pulldown keyword to create a pulldown menu.

Comments:

Pulldown menus are normally used in menu bars, and in other menus that contain actions rather than states. A pulldown menu does not have a current state; selecting a menu item normally performs an action, rather than changing the menu's current setting.

push

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

go command, ID property, lockRecent property, pop command

Summary

Places a card's long ID in a list, from which it can be retrieved later with the pop command.

Syntax

push card

Example Code

```
push card -- pushes current card  
push card "Starting Point"
```

Comments

Use the push command to store one or more locations for later use, or to keep a reference to the current card so you can return to it after temporarily visiting another card.

Parameters:

The card is any card reference.

Comments:

When you push a card, its ID is placed at the top of the list. The next time you use the pop command, that card's ID is deleted from the list, and the previously-pushed card becomes the first line of the list. In this way, you can store multiple locations with the push command, and use the pop command to revisit them in reverse order.

The list of pushed cards is cleared when you quit the application.

The Home card (the first card of the "license.rev" stack) is always the last card on the list.

put

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

after keyword, before keyword, constant command, get command, global command, libURLftpUpload command, libURLSetFTPStopTime command, local command, message box keyword, paste command, set command, How to access the Internet from behind a firewall, How to copy text between fields, How to respond to a change in field contents, Why can't I upload a file?, Why does a variable lose its value?, Recipe for building a repeated string

Summary

Places a value in a container.

Syntax

put value [{before | into | after} container]
put value into URL destinationURL

Example Code

```
put "ABC" -- puts "ABC" into message box
put 3 + 12 into myVariable -- myVariable now contains "15"
put return & someGlobalVar after field "Accumulations"
put field 1 && field 2 into URL "file:more.txt"
put homegrownMP3Data into URL "binfile:My file.MP3"
```

Comments

Use the put command to set the value of a variable, put text into a field, put data into a file, display text in the message box, or upload a file to a server.

Parameters:

The value is any expression that evaluates to a string.

The container is a field, button, image, URL, or variable, or the message box.

Comments:

If you use the put into form, the value replaces anything that was previously in the container.

The put before and put after forms place the value before or after the current contents of the container.

If you don't specify a container, the put command puts the value into the message box.

If you put a value into a nonexistent container, Revolution assumes it is a variable name, and initializes the container as a local variable. This is the most common way to create a local variable. You can also explicitly declare local variables with the local command. If the explicitVariables property is set to true, you must declare local variables before using them, and using the put command with a nonexistent variable will cause an error instead of creating the variable.

The action of the put...into URL form varies depending on the type of URL:

ĩ When used with a file or binfile URL, places the value in the specified file. If the file doesn't exist, the put command creates it. (Use the file URL type for text data and the binfile URL type for binary data.)

ĩ When used with a resfile URL, sets the resource fork of the specified file to the value. Because resource forks have a specific format, in general you should use put with a resfile URL only when the value is also a resfile URL. The following example copies the entire resource fork of one file to another:

```
put URL "resfile:My Source" into URL "resfile:Destination"
```

Note: Unlike its use with the file and binfile URL types, the put command, when used with a resfile URL, does not create the file if it doesn't exist. Instead, an error message is returned in the result. To create the file, first use a statement like the following:

```
put empty into URL "file:filePath"
```

Then you can use the put command with a resfile URL type to create the resource fork.

ĩ When used with an ftp URL, uploads the value to an FTP server. If an error occurs during uploading, the error is placed in the result function. The first word returned by the result function is "error", followed (where appropriate) by the text of the error message returned by the FTP server, including the server response code.

ĩ When used with an http URL, uses the HTTP PUT method to upload the value to the server. However, since most HTTP servers don't implement the PUT method, you usually will use an ftp URL instead to upload files to an HTTP server.

Note: Uploading a URL by putting a value into it is a blocking operation: that is, the handler pauses until Revolution is finished uploading the value. Since contacting a server may take some time due to network lag, URL operations may take long enough to be noticeable to the user. To upload without blocking, use the libURLftpUpload command instead.

Important! If a blocking operation involving a URL (using the put command to upload a URL, the post command, the delete URL command, or a statement that gets an ftp or http URL) is going on, no other blocking URL operation can start until the previous one is finished. If you attempt to use a URL in an expression, or put data into a URL, while another blocking URL operation is in progress, the result is set to "Error Previous request not completed".

QTDebugStr

message

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentTimeChanged message, hotspotClicked message, nodeChanged message, play command

Summary

Sent to a player containing a QuickTime movie when the movie performs a "DebugStr" action.

Syntax

QTDebugStr string

Example Code

```
on QTDebugStr myMessage -- display info for this
  -- section of the movie in another stack window
  go card myMessage of stack "Info"
end QTDebugStr
```

Comments

Handle the QTDebugStr message to respond to actions embedded in a QuickTime movie.

Parameters:

The string is the QuickTime parameter the movie passed to Revolution when it performed the "DebugStr" action.

Comments:

The movie author sets the string during development of the movie. When QuickTime executes the "DebugStr" action, Revolution sends the QTDebugStr message to the player, with the string the movie author specified.

QTEffects

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer effect command, hide command, QTVersion function, show command, visual effect command,
How to find out whether QuickTime is available

Summary

Returns a list of QuickTime special effects.

Syntax

the QTEffects

QTEffects()

Example Code

```
the QTEffects
if field "Effect" is not among the lines of the QTEffects then beep
```

Comments

Use the QTEffects function to find out which QuickTime special effects are currently installed, and can therefore be used with the visual effect command.

Value:

The QTEffects function returns a list of installed effect names, one per line.

Comments:

The effects returned by the QTEffects function can be used as visual effects with the visual effect, unlock screen, show, and hide commands.

QTVersion

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

dontUseQT property, play command, QTEffects function, recordFormats function, systemVersion function, How to find out whether QuickTime is available

Summary

Returns the version of QuickTime installed on the system.

Syntax

the QTVersion

QTVersion()

Example Code

the QTVersion

```
if the QTVersion < 3 then answer "You need QuickTime 3.0 or later."
```

Comments

Use the QTVersion function to make sure the version of QuickTime that's installed is recent enough to handle any version-dependent features in movies you want to play.

Value:

The QTVersion function returns a positive number.

Comments:

On Unix systems, the QTVersion function always returns 2.0.

On Mac OS and Windows systems, if QuickTime is not installed, the QTVersion function returns 0.0. Otherwise, it returns the QuickTime version number.

Tip: It can take Revolution a second or two to load the code needed to use QuickTime, depending on the machine speed. Since this code is only loaded into memory once per session, you can speed up the first occurrence of a QuickTime-related action by calling the QTVersion function during otherwise dead time—for example, during startup of your application—to preload the code.

queryRegistry

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

deleteRegistry function, lowResolutionTimers property, MCISendString function, platform function, setRegistry function

Summary

The queryRegistry function returns the value of an entry in the Windows system registry.

Syntax

the queryRegistry of keyPath

queryRegistry(keyPath[,typeVariable])

Example Code

```
queryRegistry( "HKEY_CLASSES_ROOT\rev")
queryRegistry(winFilePath,myType)
```

Comments

Use the queryRegistry function to get system settings on a Windows system.

Parameters:

The keyPath parameter is the path to a registry entry.

The typeVariable is any legal variable name.

Value:

The queryRegistry function returns a string.

Comments:

The first part of the keyPath should be one of the predefined handle values. If the keyPath ends in ", the value returned is the default value for the key.

If you specify a typeVariable, the type of the data in the registry entry is placed in that variable.

Tip: To convert binary data you get from the registry to a string, use the binaryDecode function.

On Mac OS, OS X, and Unix systems, the queryRegistry function returns "not supported".

Changes to Transcript:

The typeVariable parameter was added in version 2.0. In previous versions, the type information was not available.

quit

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close process command, kill command, shutdown message, shutdownRequest message, signal message, How to quit the application, How to respond to quitting an OS X application, Why does my application quit when I close its windows?, File menu > Quit

Summary

Quits the application.

Syntax

quit

Example Code

```
quit  
if it is "OK" then quit
```

Comments

Use the quit command to exit Revolution (or a standalone application).

Comments:

Once issued, the quit command cannot be intercepted. If you want to ask the user for confirmation or check a condition before deciding whether to quit, use an if control structure:

```
on getMeOuttaHere  
  answer "Are you sure you want to quit?" with "No way" or "OK"  
  if it is "OK" then quit  
end getMeOuttaHere
```

quote
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

& operator, && operator, constant command, How to include a quote in an expression

Summary

Equivalent to the double quote character " (ASCII 34).

Syntax

Example Code

```
if last char of it is quote then delete last char of it  
put quote & field 2 & quote into responseToPost
```

Comments

Use the quote constant to embed quotes in a string, and to refer to quotes in a script.

Comments:

You must use this constant because if you use the quote character itself, it is treated as the beginning or end of a literal string and causes an execution error.

radioBehavior

property

Synonyms

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

autoHilite property, family property, hilite property, hilitedButton property, hilitedButtonID property, hilitedButtonName property, style property, How to create a radio button cluster, How to change the selected button in a radio button cluster, How to give a border to a radio button cluster

Summary

Specifies that only one radio button in a group can be highlighted at a time.

Syntax

set the radioBehavior of group to {true | false}

Example Code

```
set the radioBehavior of last group to true
```

Comments

Use the radioBehavior property to create a radio-button cluster.

Value:

The radioBehavior of a group is true or false.

By default, the radioBehavior property of newly created groups is set to true.

Comments:

If a group's radioBehavior property is set to true, highlighting any radio button in the group unhighlights any other radio buttonsóthat is, the buttons in the group behave like a radio-button cluster, with the options being mutually exclusive.

If the radioBehavior is false, highlighting a button does not affect the hilite property of other buttons in the group.

The radioBehavior property does not affect buttons whose style is not set to radioButton.

raiseMenus

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

menubar property, pointerFocus property, raisePalettes property

Summary

Specifies whether the window with the menu bar comes to the front when the user opens a menu.

Syntax

set the raiseMenus to {true | false}

Example Code

```
set the raiseMenus to true
```

Comments

Use the raiseMenus property to control the behavior of windows.

Value:

The raiseMenus is true or false.

By default, the raiseMenus property is set to false.

Comments:

On some Unix systems, setting this property to true may cause menus to appear under the window.

The setting of this property has no effect on Mac OS systems.

raisePalettes

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

activatePalettes property, hidePalettes property, palette command, style property, How to keep a window on top of other windows

Summary

Specifies whether palettes always float above standard windows.

Syntax

set the raisePalettes to {true | false}

Example Code

```
set the raisePalettes to true
```

Comments

Use the raisePalettes property to control interleaving of palettes with other windows.

Value:

The raisePalettes is true or false.

By default, the raisePalettes property is set to true.

Comments:

If the raisePalettes property is true, palettes float in their own layer above standard windows. In other words, all palettes are always in front of editable windows and modeless dialog boxes if this property is set to true.

If it is set to false, palette windows can be interleaved with standard windows.

Important! Setting this property to true may result in window flashing on Unix systems.

random

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

any keyword, randomSeed property, round function, How to shuffle the items or lines in a container, Recipe for checking whether a volume or folder is write-protected

Summary

Returns a random integer.

Syntax

the random of upperLimit
`random(upperLimit)`

Example Code

```
random(22) -- returns a number between 1 and 22  
random(item 1 of field "Numbers")  
sort lines of myVar by random(the number of lines of myVar)
```

Comments

Use the random function to pick a random member of a set, or to generate a random number.

Parameters:

The upperLimit is a number.

Value:

The random function returns an integer.

Comments:

If the upperLimit is a positive integer, the random function returns an integer between 1 and the upperLimit. If the upperLimit is a number that is not an integer, the random function returns an integer between 1 and round(upperLimit).

To generate a random number between two integers, use a handler like this:

```
function randomInRange lowerLimit,upperLimit
```

```
    return random(upperLimit - lowerLimit + 1) + lowerLimit - 1  
end randomInRange
```

randomSeed

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

any keyword, random function

Summary

Determines the seed value used to generate random numbers.

Syntax

set the randomSeed to integer

Example Code

```
set the randomSeed to 4570422
```

Comments

Use the randomSeed property to ensure greater randomness when generating random numbers.

Value:

The randomSeed property is an integer.

Comments:

Changing the randomSeed property changes the pseudorandom numbers generated by the random function and used by the any keyword. Using the same seed creates the same sequence of pseudorandom values.

For example, if you call the random function five times to generate a list of five numbers, then change the randomSeed to another value, the next five calls to the random function will produce a different list of five numbers. However, if you set the randomSeed back to its original value and call the random function five more times, the list of five numbers is the same as the first list you generated.

Revolution uses a new randomSeed every time the application is started up.

rawKeyDown

message

Synonyms

Objects

field, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

capsLockKey function, commandKey function, keyDown message, keysDown function, mouseDown message, optionKey function, rawKeyUp message, shiftKey function, type command

Summary

Sent when the user presses any key.

Syntax

rawKeyDown keyCode

Example Code

```
on rawKeyDown theKeyNumber
  if theKeyNumber is 65308 then increaseScroll -- mouse wheel down
  else if theKeyNumber is 65309 then decreaseScroll -- mouse wheel up
  else pass rawKeyDown -- don't forget this!
end rawKeyDown
```

Comments

Handle the rawKeyDown message if you want to intercept raw events from the keyboard, or from a mouse wheel, or if you want to handle keys that aren't mapped to any character.

Parameters:

The keyCode is the raw keycode of the pressed key.

Comments:

If the rawKeyDown handler does not pass the message or send it to a further object in the message path, the keypress has no effect. Passing the message allows the keypress to have its normal effect.

If the rawKeyDown message is sent as the result of a keypress, the message is sent to the active (focused) control, or to the current card if no control is focused.

A rawKeyDown message is also sent when the user moves the mouse wheel on a scrolling mouse; in this case, the message is sent to the control under the mouse pointer.

If the insertion point is in a field, the entry of typed characters is triggered by the `rawKeyDown` message. This means that trapping the `rawKeyDown` message and not passing it prevents typing from being entered in the field.

Cross-platform note: On Mac OS systems, no message is sent when a modifier key (Shift, Option, Control, or Command) is pressed, unless another key is pressed along with the modifier key. Mouse wheels do not send a `rawKeyDown` message on Mac OS systems.

rawKeyUp

message

Synonyms

Objects

field, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

capsLockKey function, commandKey function, keysDown function, keyUp message, mouseDown message, optionKey function, rawKeyDown message, shiftKey function, type command

Summary

Sent when the user releases any key that was pressed (other than a modifier key).

Syntax

rawKeyUp keyCode

Example Code

```
on rawKeyUp theCode -- boldface all asterisks
  get the HTMLText of me
  replace "*" with "<b>*</b>" in it
  set the HTMLText of me to it
end rawKeyUp
```

Comments

Handle the rawKeyUp message if you want to intercept raw events from the keyboard or if you want to handle keys that aren't mapped to any character.

Parameters:

The keyCode is the raw keycode of the released key.

Comments:

If the rawKeyUp message is sent as the result of a keypress, the message is sent to the active (focused) control, or to the current card if no control is focused.

If the message is sent as the result of typing something in a field, the rawKeyDown message is sent before the text is changed, and the rawKeyUp message is sent after the change has been made.

read from driver command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close driver command, driverNames function, int1 keyword, int2 keyword, int4 keyword, open driver command, read from file command, read from process command, read from socket command, real4 keyword, real8 keyword, uInt1 keyword, uInt2 keyword, uInt4 keyword, write to driver command, Why do some characters change when transferred between systems?, Why is there a problem with line endings?

Summary

Takes incoming data from a device that has been opened with the open driver command, and places the data in the it variable.

Syntax

read from driver deviceName [at start]

{until {string | end | EOF | empty} | for amount [chunkType]} [in time]

Example Code

```
read from driver ".BIn" for 3 lines
read from driver it for 2 chars in 2 seconds
```

Comments

Use the read from driver command to get data from a peripheral device such as a modem.

Parameters:

The deviceName is the name of a device driver that's installed on the system and that you have previously opened with the open driver command.

The start specifies the character or byte position in the device's input where you want to begin reading.

The string is any expression that evaluates to a string. When Revolution encounters the string in the data coming in from the device, it stops reading. If the string is not encountered, the read from driver command continues reading as long as there is data to be read.

The amount is a positive integer and specifies how much data to read.

The chunkType is one of chars, characters, words, items, lines, int1, uInt1, int2, uint2, int4, or uint4. The read from driver command reads amount of the specified chunkType. If you don't specify a chunkType, amount characters are read.

The time is the time to wait for the read to be completed, in milliseconds, seconds, or ticks.

Comments:

The device to read from must be opened first with the open driver command, and the mode the device was opened in must be either read or update. If the device is not open or is open write-only, the result function is set to "File is not open for read."

The until string form reads data until the specified string is encountered. The until empty, until end, and until EOF forms read data until there is no more data to be read.

The data is placed in the it variable after reading. If you specified a binary data chunkType (int1, uInt1, int2, uint2, int4, or uint4), the data consists of a comma-separated list of numbers, one for the numerical value of each chunk that was read. Otherwise, the data is placed in the it variable as it appears in the output.

If you specify a time and the read is not completed when that time has elapsed, the result function is set to "time out". If the read was successful, the result is set to empty.

Changes to Transcript:

Support for using serial drivers with OS X systems was added in version 2.0.

read from file

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

close file command, EOF constant, get command, int1 keyword, int2 keyword, int4 keyword, open file command, openFiles function, read from driver command, read from process command, read from socket command, real4 keyword, real8 keyword, return constant, seek command, stdin keyword, sysError function, uInt1 keyword, uInt2 keyword, uInt4 keyword, write to file command, About filename specifications and file paths, Why do some characters change when transferred between systems?, Why is there a problem with line endings?

Summary

Takes data from a file that has been opened with the open file command, and places the data in the it variable.

Syntax

read from {file pathName | stdin} [at start]

{until {string | end | EOF | empty} | for amount [chunkType]} [in time]

Example Code

```
read from file "Test" for 8 -- reads 8 characters
read from file "COM1:" at 20 until EOF
read from file (field "Datafile") at -100 for charsToRead
read from stdin for 1 line
```

Comments

Use the read from file command to get data from a file.

Parameters:

The pathName specifies the name and location of the file you want to read from. It must be the same as the path you used with the open file command.

Important! The pathName is case-sensitive, even on platforms where file names are not case-sensitive. It must be exactly the same—including the case of characters—as the name you used with the open file command.

If you specify the name of a serial port on Mac OS or Windows systems, Revolution reads from the specified port. The names of serial ports end in a colon (:).

The start specifies the character or byte position in the file where you want to begin reading. A positive number begins start characters after the beginning of the file; a negative number begins start characters before the end of the file.

The string is any expression that evaluates to a string. When Revolution encounters the string in the file, it stops reading. If the string is not encountered, the read from file command continues reading until it reaches the end of the file.

If you specify any of EOF, end, or empty, the read continues reading until it reaches the end of the file. (If you're reading from a serial port, you must use the form read from file portname until empty.)

The amount is a positive integer and specifies how much data to read.

The chunkType is one of chars, characters, words, items, lines, int1, uInt1, int2, uint2, int4, or uint4. The read from file command reads amount of the specified chunkType. If you don't specify a chunkType, amount characters are read from the file.

Comments:

The file to read from must be opened first with the open file command, and the mode the file was opened in must be either read or update. If the file is not open or is open write-only, the result function is set to "File is not open for read."

If you don't specify a start, Revolution begins at the position determined by the seek command, or wherever the last read from file or write to file command to that file left off, or at the first character, if the file has not been accessed since being opened, or at the last character, if the file was opened in append mode.

The until string form reads data until the specified string is encountered. The until end, until EOF, and until empty forms are synonyms, and read data up to the end of the file. You can read an entire file by opening it and reading until the end:

```
open file fileToRead
read from file fileToRead until EOF
close file fileToRead
```

The read from stdin form reads from the standard input (on Unix systems). The standard input is always open, so you can read from it without first opening it.

The data is placed in the it variable after reading. If you specified a binary data chunkType (int1, uInt1, int2, uint2, int4, or uint4), the data consists of a comma-separated list of numbers, one for the numerical value of each chunk that was read. Otherwise, the data is placed in the it variable as it appears in the file.

If the read from file command encounters the end of the file, the result function is set to "eof". If the command was successful and did not encounter the end of the file, the result function is set to empty.

Tip: As an alternative to the `open file` and `read from file` commands, you can also use the `URL` keyword with `get` and other commands to access the contents of a file.

read from process command

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close process command, EOF constant, int1 keyword, int2 keyword, int4 keyword, open process command, openProcesses function, read from driver command, read from file command, read from socket command, real4 keyword, real8 keyword, return constant, uInt1 keyword, uInt2 keyword, uInt4 keyword, write to process command, Why do some characters change when transferred between systems?, Why is there a problem with line endings?

Summary

Accepts the output of a process that was opened with the open process command, placing the data in the it variable.

Syntax

read from process processName [at start]

{until {string | empty | end | EOF} | for amount [chunkType]} [in time]

Example Code

```
read from process "compileData" for 20
read from process "/etc/pr" at 2 until linefeed -- skip 2 chars
read from process myProcess for 10 int4s in 250 milliseconds
```

Comments

Use the read from process command to get the output data from another program.

Parameters:

The processName specifies the name and location of the process you want to read from. It must be the same as the path you used with the open process command.

Important! The processName is case-sensitive, even on platforms where file names are not case-sensitive. It must be exactly the same—including the case of characters—as the name you used with the open process command.

The start specifies the character or byte position in the process output where you want to begin reading.

The string is any expression that evaluates to a string. When Revolution encounters the string in the process output, it stops reading. If the string is not encountered, the read from process command continues reading as long as there is data to be read.

The amount is a positive integer and specifies how much data to read.

The chunkType is one of chars, characters, words, items, lines, int1, uInt1, int2, uint2, int4, or uint4. The read from process command reads amount of the specified chunkType. If you don't specify a chunkType, amount characters are read.

The time is the time to wait for the read to be completed, in milliseconds, seconds, or ticks.

Comments:

The process to read from must be opened first with the open process command, and the mode the process was opened in must be read or update. If the process is not running or is write-only, the result function is set to "Process is not open for read."

The until string form reads data until the specified string is encountered. The until empty, until end, and until EOF forms read data until the EOF character is encountered in the data stream.

The data is placed in the it variable after reading. If you specified a binary data chunkType (int1, uInt1, int2, uint2, int4, or uint4), the data consists of a comma-separated list of numbers, one for the numerical value of each chunk that was read. Otherwise, the data is placed in the it variable as it appears in the output.

If the read from process command encounters the end of the data output, the result function is set to "eof". If you specify a time and the read is not completed when that time has elapsed, the result function is set to "time out". If the read was successful, the result is set to empty.

Changes to Transcript:

Support for using the read from process command on OS X systems was added in version 2.0.

read from socket command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

accept command, close socket command, int1 keyword, int2 keyword, int4 keyword, open socket command, read from file command, read from driver command, read from process command, real4 keyword, real8 keyword, socketError message, socketTimeout message, uInt1 keyword, uInt2 keyword, uInt4 keyword, write to socket command, Why do some characters change when transferred between systems?, Why is there a problem with line endings?

Summary

Accepts data from a socket and places the data in the it variable.

Syntax

read from socket socketID [until string | for amount [chunkType]]

[with message callbackMessage]

Example Code

```
read from socket "www.example.net:80" -- reads next character
read from socket "www.example.net:80" for 50 -- reads next 50 chars
read from socket "127.0.0.0:20|foo" until linefeed
read from socket mySocket for 30 uInt2s with message "gotData"
```

Comments

Use the read from socket command to get data from another system via a TCP socket.

Parameters:

The socketID is the identifier (set when you opened the socket) of the socket you want to get data from.

The socket identifier starts with the IP address of the host the socket is connected to, and may optionally include a port number (separated from the IP address by a colon). If there is more than one socket connected to that host and port, you can specify which socket by appending the connection name or number that was assigned when the socket was opened, separated from the port number by a vertical bar (|).

The string is any expression that evaluates to a string. When Revolution encounters the string in the socket data, it stops reading. If the string is not encountered, the read from socket command continues reading as long as there is data to be read.

The amount is a positive integer and specifies how much data to read.

The chunkType is one of chars, characters, words, items, lines, int1, uInt1, int2, uint2, int4, or uint4. The read from socket command reads amount of the specified chunkType. If you don't specify a chunkType, amount characters are read.

The callbackMessage is the name of a message to be sent when the read is successfully completed.

Comments:

The socket to read from must be opened first with the open socket command. If the socket is not open, the result function is set to "Socket is not open."

The until string form reads data until the specified string is encountered. The for amount form reads data until the specified number of chunks have arrived. If you don't specify a string or amount, the read from socket command reads the next character of data from the socket.

If you specify a callbackMessage, the message is sent to the object whose script contains the read from socket command, as soon as the read is finished. This message has two parameters: the socketID and the data received from the socket.

If you don't specify a callbackMessage, the handler pauses until the read has been completed, or until the time set in the socketTimeoutInterval property has passed. The data is placed in the it variable after reading. If you specified a binary data chunkType (int1, uInt1, int2, uint2, int4, or uint4), the data consists of a comma-separated list of numbers, one for the numerical value of each chunk that was read. Otherwise, the data is placed in the it variable as it appears in the incoming data stream.

real4
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

int4 keyword, read from file command, read from process command, real8 keyword, uInt4 keyword, write to file command, write to process command

Summary

Used with the read from file, read from process, and read from socket commands to signify a chunk of binary data the size of a signed 4-byte real.

Syntax

Example Code

```
read from file "image.gif" for 3 real4
```

Comments

Use the real4 keyword to read data from a binary file.

Comments:

If you specify real4 as the chunk type in a read from file, read from process, or read from socket command, the data is returned as a series of numbers separated by commas, one for each chunk read.

real8

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

int4 keyword, read from file command, read from process command, real4 keyword, uint4 keyword, write to file command, write to process command

Summary

Used with the read from file, read from process, and read from socket commands to signify a chunk of binary data the size of a signed 8-byte real.

Syntax

Example Code

```
read from process currentInput for 1 real8  
read from file "Thing.jpg" for 12 real8
```

Comments

Use the real8 keyword to read data from a binary file.

Comments:

If you specify real8 as the chunk type in a read from file, read from process, or read from socket command, the data is returned as a series of numbers separated by commas, one for each chunk read.

recent
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

go command, About object types and object references

Summary

Used in card references.

Syntax

Example Code

```
go recent card  
put field "Type" of recent card into lastTypeChecked
```

Comments

Use the recent keyword to refer to the previously-visited card.

Comments:

Whenever you switch cards, unless the lockRecent property is set to true, each card you've visited is added to the recent cards list. In card references, the recent keyword specifies the card most recently visited.

recentCards

property

Synonyms

Objects

stack, global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

go command, lockRecent property, recentNames property, View menu > Go Recent, Shortcut to review the recent cards list

Summary

Reports the long ID properties of recently visited cards.

Syntax

get the recentCards [of stack]

Example Code

```
do "go" && line 2 of the recentCards -- equivalent to "go recent 2"
```

Comments

Use the recentCards property to find out which cards the user has visited, or to re-visit a card.

Value:

The recentCards is a list of long ID properties of cards, one ID per line. If a stack is specified, the list contains only the recent cards in the specified stack. If not, all recent cards are listed.

Comments:

The recentCards are listed in reverse order, with the current card at the top of the list, the most recently visited card next, and so on. Each visit to a card causes its ID to be placed at the top of the list, even if the same card appears elsewhere in the list. The recentCards thus provides a complete track of the visitor's navigation.

Cards visited while the lockRecent property is set to true are not added to the list.

recentNames

property

Synonyms

backList

Objects

stack, global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

go command, lockRecent property, View menu > Go Recent, Shortcut to review the recent cards list

Summary

Reports a list of the most recently visited cards, in reverse order.

Syntax

get the recentNames [of stack]

Example Code

```
put line 1 of the recentNames after myVisitedCards
```

Comments

Use the recentNames property to find out which cards the user has visited, or to re-visit a card.

Value:

The recentNames is a list of short name properties of cards, one name per line. If a stack is specified, the list contains only the recent cards in the stack. If not, all recent cards are listed.

Comments:

The recentNames are listed in reverse order, with the current card at the top of the list, the most recently visited card next, and so on. Each visit to a card causes its name to be placed at the top of the list, even if the same card appears elsewhere in the list. The recentNames thus provides a complete track of the visitor's navigation.

Cards visited while the lockRecent property is set to true are not added to the list.

record sound

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer record command, play command, recordChannels property, recordCompression property, recordFormat property, recordInput property, recordLoudness property, recordRate property, recordSampleSize property, recording property, stop recording command, How to find out whether QuickTime is available

Summary

Starts recording from the system's audio input to a file.

Syntax

record sound file filePath

Example Code

```
record sound file "Testing"  
record sound file it
```

Comments

Use the record sound command to record the user's speech, import sound data from a CD into an audio file, or record music from an external microphone.

Parameters:

The filePath is the name and location of the file that will hold the recorded sound data.

Comments:

The record sound command uses QuickTime to record.

While recording is going on, the recording property is set to true. To stop recording, use the stop recording command or set the recording to false.

The technical specifications of the recording are determined by the settings of the recordCompression, recordChannels, recordRate, and recordSampleSize properties. The file format is determined by the recordFormat property.

Important! If the recordFormat is set to "movie", the resulting file is in the QuickTime file format. QuickTime files cannot be played as audio clips. To play such a sound, either create a player and set its filename property to the path of the file you recorded, or use the play videoclip form of the play command.

Changes to Transcript:

The syntax of the record sound command was simplified in version 2.0. In previous versions, specification of sample rate, compression format, and whether to display a settings dialog box was included in the syntax for the record sound command. These capabilities are now available using the answer record command and the recordRate and recordCompression properties.

The ability to choose the format of the sound file with the recordFormat property was introduced in version 2.0. In previous versions, the record sound command always produced a QuickTime file.

recordChannels

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer record command, record sound command, recordInput property

Summary

Specifies whether to record sound in stereo or mono.

Syntax

set the recordChannels to [1 | 2]

Example Code

```
set the recordChannels to 1 -- record in mono
if the recordChannels is 2 then hilite button "Stereo"
```

Comments

Use the recordChannels property to trade off between sound quality and disk space when you record a sound.

Value:

The recordChannels is either 1 or 2.

By default, the recordChannels property is set to 1 (monophone).

Comments:

The recordChannels property determines the kind of recording made when you use the record sound command to record a sound.

Note: The answer record command sets the recordChannels property according to what the user chose in the answer record dialog box.

recordCompression

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer record command, record sound command, recordCompressionTypes function, recordFormat property, recordRate property, recordSampleSize property

Summary

Specifies which codec is used to compress recorded sounds.

Syntax

set the recordCompression to codecName

Example Code

```
set the recordCompression to "MAC6"  
if the recordCompression is "raw " then requestAnotherType
```

Comments

Use the recordCompression property to trade off between sound quality and disk space when you record a sound.

Value:

The recordCompression is a four-character string.

By default, the recordCompression property is "raw ".

Comments:

The recordCompression property determines what kind of compression is used when you use the record sound command to record sound to a file.

Not all codecs are compatible with all supported sound file formats. If you try to record sound using a recordCompression that is not compatible with the file format specified by the recordFormat property, the compression type will be changed automatically to a codec that is compatible with the specified recordFormat.

The codecs are installed by QuickTime. You can obtain a list of available codecs using the `recordCompressionTypes` function. The default setting, "raw " (note the trailing space), indicates no compression will be used.

Note: The `answer record` command sets the `recordCompression` property according to what the user chose in the answer record dialog box.

recordCompressionTypes

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer record command, dontUseQT property, record sound command, recordCompression property

Summary

Returns a list of audio codecs supported by the currently installed version of QuickTime.

Syntax

the recordCompressionTypes

recordCompressionTypes()

Example Code

```
the recordCompressionTypes
if storedType is not among the lines of the recordCompressionTypes
    then answer record
```

Comments

Use the recordCompressionTypes function to find out which formats you can use to record sound with the record sound command.

Value:

The recordCompressionTypes function returns a list of available audio codecs, one per line. Each line consists of two items:

- the codec's name

- the codec's four-character identifier

Comments:

You specify a codec by setting the recordCompression property to one of the four-character identifiers returned by the recordCompressionTypes. The record sound command then uses that codec to compress recorded sounds. Each codec supports a different type of compression.

The recordCompressionTypes function requires QuickTime to be installed. If QuickTime is not installed, the recordFormats function returns empty.

Tip: It can take Revolution a second or two to load the code needed to use QuickTime, depending on the machine speed. Since this code is only loaded into memory once per session, you can speed up the first occurrence of a QuickTime-related action by calling the QTVersion function during otherwise dead time—for example, during startup of your application—to preload the code.

recordFormat

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

play command, record sound command, recordCompression property

Summary

Specifies the file format for sound files recorded with the record sound command.

Syntax

set the recordFormat to fileFormat

Example Code

```
set the recordFormat to "wave"  
set the recordFormat to the storedSoundFormat of me
```

Comments

Use the recordFormat property to specify what kind of file to create when recording a sound, depending on how the file will be used.

Value:

The recordFormat is one of the following:

aiff	records a file in AIFF format
wave	records a file in WAV format
ulaw	records a file in AU format
movie	records a file in QuickTime format

By default, the recordFormat property is "aiff".

Comments:

The recordFormat property determines what kind of file is created when you use the record sound command to record sound.

Not all sound file formats are compatible with all supported codecs. If you try to record sound using a `recordCompressionType` that is not compatible with the `recordFormat`, the compression type will be changed automatically to a codec that is compatible with the specified `recordFormat`.

Important! If the `recordFormat` is set to "movie", the resulting file is in the QuickTime file format. QuickTime files cannot be played as audio clips. To play such a sound, either create a player and set its `filename` property to the path of the file you recorded, or use the `play videoclip` form of the `play` command.

recordFormats

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

QTVersion function, record sound command, recordCompressionTypes command

Summary

Removed in version 2.0.

Syntax

the recordFormats

recordFormats()

Example Code

Comments

The recordFormats function reported information about the most recent execution error.

For the information formerly returned by the recordFormats function, see the recordCompressionTypes function.

recording

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

record sound command, sound function, stop recording command

Summary

Reports whether a sound is currently being recorded.

Syntax

set the recording to {true | false}

Example Code

```
set the recording to false
wait until the recording is true
```

Comments

Use the recording property to determine whether a recording is being made or to stop recording.

Value:

The recording property is true or false.

Comments:

The recording property is true if sound is currently being recorded with the record sound command, false if it is not.

If a recording is currently being made, you can set the recording property to false to stop recording. This is equivalent to using the stop recording command.

recordInput

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

open driver command, playDestination property, record sound command, recordChannels property

Summary

Specifies where the record sound command should listen when recording a sound file.

Syntax

set the recordInput to soundSource

Example Code

```
set the recordInput to "imic" -- internal microphone
set the recordInput to "cd "
if the recordInput is not in permittedDevices then exit mouseUp
```

Comments

Use the recordInput property to record sound from the system's microphone, a CD player, or other sources.

Value:

The recordInput is a four-character string.

The default value for the recordInput property is "dflt".

Comments:

The recordInput property determines which input device is used as the sound source when you use the record sound command to record sound.

The value "dflt" indicates that the record sound command should use the sound input device chosen in the user's system settings. If a different soundSource is specified, that input device is used instead.

The possible soundSources vary, depending on the QuickTime version installed and on the system's hardware configuration. QuickTime 3.0 and later supports the following:

- imic records from the internal microphone

emic	records from the external sound input jack
cd	records from an internal CD player
irca	records from an RCA input jack
tvfm	records from an FM radio tuner
idav	records from a DAV analog input port
mbay	records from a media-bay device
modm	records from the modem
zvpc	records from zoom video input

recordLoudness

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

beepLoudness property, playLoudness property, record sound command, recording property

Summary

Returns the volume of the sound currently being recorded.

Syntax

the recordLoudness

recordLoudness()

Example Code

```
the recordLoudness
if the recordLoudness is zero then pauseRecording
```

Comments

Use the recordLoudness function to determine whether a sound is currently being made while recording is taking place, or to provide feedback on the input sound level.

Value:

The recordLoudness function returns a positive integer.

Comments:

The recordLoudness function returns zero if no recording is currently taking place (that is, if the recording property is false).

recordRate

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer record command, record sound command, recordInput property, recordSampleSize property

Summary

Specifies how often the sound input should be read during sound recording.

Syntax

set the recordRate to kSamplesPerSecond

Example Code

```
set the recordRate to 48 -- sample at 48 kHz
```

Comments

Use the recordRate property to trade off between sound quality and disk space when you record a sound.

Value:

The recordRate is a non-negative number.

By default, the recordRate property is 22.05.

Comments:

The recordRate property determines the sound sampling rate when you use the record sound command to record sound to a file.

The possible recordRates depend on the system's capabilities.

The kSamplesPerSecond is the number of thousands of times per second the sound is sampled. The higher this number, the higher the quality of the resulting sound, and the larger the resulting file. For best sound quality, the higher recordRates should be used when the sound input is of high quality—for example, when recording from a CD and when music is being recorded. (44.1 kHz is "CD-quality".) In general, you can use lower recordRates for spoken words than for music without compromising sound quality.

Note: The answer record command sets the recordRate property according to what the user chose in the answer record dialog box.

recordSampleSize

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer record command, recordCompression property, recordRate property

Summary

Specifies how large each chunk of data should be when recording sound.

Syntax

set the recordSampleSize to sizeInBits

Example Code

```
set the recordSampleSize to 16
```

Comments

Use the recordSampleSize property to trade off between sound quality and disk space when you record a sound.

Value:

The recordSampleSize is a positive integer.

By default, the recordSampleSize property is set to 8.

Comments:

The recordSampleSize property determines the sound sample size when you use the record sound command to record sound to a file.

The possible recordSampleSize depend on the system's capabilities, but are typically 8 and 16.

Note: The answer record command sets the recordSampleSize property according to what the user chose in the answer record dialog box.

rectangle

keyword

Synonyms

rect

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

brushColor property, brushPattern property, choose command, lineSize property, penColor property, style property, tool function, Object menu > New Control > Rectangle Button, Object menu > New Control > Rectangle Graphic

Summary

Specifies that a button, field, or graphic object is shaped like a rectangle. It also designates the paint tool used to draw rectangles and squares.

Syntax

Example Code

```
choose rectangle tool  
set the style of button ID 3 to rectangle
```

Comments

Use the rectangle keyword to paint a rectangle or square with the penColor or to change a graphic to a rectangle shape.

Comments:

When using the Rectangle tool, the cursor is a crosshairs shape (over images) or an arrow (anywhere else). This tool is used to draw a rectangle or square in the penColor, filled with the brushColor (or brushPattern).

If you try to paint with the Rectangle tool on a card that has no images, an image the size of the card is created to hold the painting. If the current card already has one or more images, painting outside the image has no effect.

Setting the style of a graphic to rectangle makes the graphic into a rectangle. Rectangle graphics, unlike painted rectangles, can be changed and reshaped: use the height and width properties to change the shape and size of the rectangle.

Setting the style of a field or button to rectangle makes it a rectangle shape.

rectangle

property

Synonyms
rect

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

bottom property, bottomLeft property, bottomRight property, crop command, editMenus property, effective keyword, height property, is within operator, left property, location property, margins property, revCacheGeometry command, revChangeWindowSize command, revUpdateGeometry command, right property, screenRect function, top property, topLeft property, topRight property, width property, windowBoundingRect property, within function, How to automatically adjust objects when a window is resized, How to change the size and position of the video grabber window, How to create a window the same size as the screen, How to determine the dimensions of an image, How to prevent the user from resizing a window, Why aren't window positions saved in my application?, Why does the window change size when I create a card?

Summary
Specifies the area within which an object is drawn.

Syntax
set the rectangle of object to left,top,right,bottom
get the [effective] rectangle of object

Example Code
set the rectangle of button "Tangle" to 20,20,45,200
set the rectangle of group 1 to the rectangle of this card

Comments
Use the rectangle property to find out how far an object extends, to move it, or to resize it.

Value:
The rectangle of an object consists of four integers separated by commas.

Comments:
The four items of an object's rectangle describe the object's left, top, right, and bottom edges:

• The left is the number of pixels between the left edge of the stack window and the leftmost pixel of the object.

ï The top is the number of pixels between the top edge of the stack window and the topmost pixel of the object.

ï The right is the horizontal distance in pixels between the left edge of the stack window and the rightmost pixel of the object.

ï The bottom is the vertical distance in pixels between the top edge of the stack window and the bottommost pixel of the object.

Note: The sides of an object's rectangle specify the lines between pixels, not the pixels themselves. For example, if an object's rectangle is "0,0,2,2", the object includes four pixels, starting at the top left corner of the card. In the case of a line or curve graphic, the graphic's rectangle encloses all the pixels in the graphic's points property without touching any of them.

If the object is a stack, its rectangle is relative to the left and top of the screen, rather than the left and top of the stack window.

The first two items of a card's rectangle are always zero. The third item is the height of the card, and the fourth is the width of the card.

Note: The rectangle of a graphic is drawn around all its points without touching them. (Usually, this makes no difference, but in some circumstances where you need to place a graphic's vertex precisely with respect to another object's rectangle, you may need to take this into account.)

If you specify the effective keyword, the rectangle includes the outline added by the showFocusBorder property. It also includes the heavy outline added to the defaultButton. If the showFocusBorder of the object is false, or the object is not currently focused, the effective rectangle is the same as the rectangle.

Changes to Transcript:

The use of the effective keyword with the rectangle property was introduced in version 1.1. In previous versions, the rectangle of the defaultButton included the heavy outline.

recursionLimit

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

Summary

Specifies how many levels deep a recursive function is allowed to go.

Syntax

set the recursionLimit to stackSizeInBytes

Example Code

```
set the recursionLimit to 20
```

Comments

Use the recursionLimit property to quickly find possible recursion bugs or to extend the ability to use deeply recursive routines.

Value:

The recursionLimit is a positive integer.

By default, the recursionLimit property is set to 1000000.

Comments:

The stackSizeInBytes specifies the CPU call stack size. ("Stack" in this sense has nothing to do with Revolution stack objects; it refers to a type of data structure used by the processor.)

The relationship between the recursionLimit and the number of levels of nesting permitted for a recursive call depends on a number of factors, including the processor type and the number of parameters passed during each function call. This means that the same recursionLimit value may allow a different maximum level of nesting, depending on the platform.

Recursion may be used deliberately, especially in processing large data sets. To allow deeper levels of recursion than usual, increase the recursionLimit.

Recursion may also occur accidentally. In this case, the recursion is usually infinite—that is, the function will keep recursing until it reaches the limit and causes an execution error. To track down such problems more quickly, reduce the `recursionLimit`. The lower the limit, the more quickly a buggy routine will cause an execution error.

redo
command

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

undo command

Summary

The redo command is not implemented and is reserved.

Syntax

Example Code

Comments

regular

keyword

Synonyms

regular polygon, reg polygon, regular poly, reg poly

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

brushColor property, brushPattern property, choose command, lineSize property, polygon keyword, polySides property, tool function, Object menu > New Control > Regular Polygon Graphic

Summary

Designates the paint tool used to draw regular polygons. It also specifies, through the style property, that a graphic is a regular polygon.

Syntax

Example Code

```
set the style of graphic "All Angles" to regular
choose regular polygon tool
```

Comments

Use the regular keyword to paint a regular polygon with the penColor or to change a graphic to a regular polygon shape.

Comments:

Regular polygons are geometric shapes whose sides are all the same length, such as squares, equilateral triangles, hexagons, and so on. (You set the number of sides with the polySides property.)

When using the Regular tool, the cursor is a crosshairs shape (over images) or an arrow (anywhere else). This tool is used to draw a polygon in the penColor, filled with the brushColor (or brushPattern). The number of sides is determined by the global polySides property.

If you try to paint with the Regular tool on a card that has no images, an image the size of the card is created to hold the painting. If the current card already has one or more images, painting outside the image has no effect.

Setting the style of a graphic to "regular" makes the graphic into a regular polygon. Regular polygon graphics, unlike painted polygons, can be changed and reshaped: use the polySides properties to change the shape.

Important! You can use the synonyms `regular polygon`, `reg polygon`, `regular poly`, and `reg poly` to choose the regular paint tool, but not to set a graphic's style property.

relative

keyword

Synonyms
rel

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

globalLoc function, localLoc function, move command, seek command

Summary

Used with the seek and move command to start the action at the current position.

Syntax

Example Code

```
seek to relative -20 in file "Input Date"  
move image "Butterfly" relative 10,-10 - 10 pixels right and up
```

Comments

Use the relative keyword with the read from file and write to file commands to move within a file, or with the move command to move a control relative to its current location.

Comments:

When used with the seek command, the relative keyword indicates how far to move the current position. If the number is positive, the seek command moves the current position by the specified number of characters forward in the file. If the number is negative, the current position is moved backward in the file.

(The current position in a file is set by the most recent read from file, write to file, or seek command. If none of these commands has been used on the file since it was opened, the current position is zero. The next read from file or write to file command starts from the new current position.)

When used with the move command, the relative keyword indicates how many pixels to the right and down to move the control.

relativePoints

property

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

dashes property, endArrow property, markerPoints property, points property, startArrow property

Summary

Specifies where the vertexes of a curve or irregular polygon graphic are located, relative to the graphic's rectangle.

Syntax

set the relativePoints of graphic to vertexList

Example Code

```
set the relativePoints of graphic "Arrows" to field "Endpoints"
```

Comments

Use the relativePoints property to find the shape of a curve or irregular polygon graphic, or to reshape the graphic.

Value:

The relativePoints of a graphic is a list of points (two integers separated by a comma), one per line.

Comments:

Each point consists of the horizontal distance in pixels from the left edge of the graphic to the vertex, a comma, and the vertical distance in pixels from the top edge of the graphic to the vertex. (The points property is the same list, but measured from the top left of the stack window rather than the top left of the graphic.)

The first line in the list is the location of the graphic's starting point. A blank line in the relativePoints indicates that the previous and next vertexes are not connected by a line—that is, the polygon or curve is broken into two (or more) pieces.

If the style property of the graphic is not polygon or curve, the setting of its relativePoints property has no effect.

relayerGroupedControls

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

editBackground property, group command, layer property, start editing command

Summary

Specifies whether you can change the layer of controls in a group even if not in group-editing mode.

Syntax

set the relayerGroupedControls to {true | false}

Example Code

```
set the relayerGroupedControls to true
```

Comments

Use the relayerGroupedControls property to change the layer of grouped controls without being in group-editing mode, or to move controls out of a group by changing their layer.

Value:

The relayerGroupedControls is true or false.

By default, the relayerGroupedControls property is set to false.

Comments:

The layer of a control is its order on the card. If two controls overlap, the one whose layer is higher covers the one whose layer is lower.

If the relayerGroupedControls is false, you can change the layer of a control that's part of a group only when editing the group. If the relayerGroupedControls is true, you can change the layer of a grouped control at any time.

Important! It is not possible for other controls to be interspersed between the controls in a single group, so changing a grouped control's layer may change its group membership if the relayerGroupedControls is true:

ï You can move a control out of a group by setting the control's layer to a number greater than the topmost control in the group, or less than the bottom-most control in the group.

ï Conversely, you can move a control into a group by setting the control's layer to a number between the bottom-most and topmost controls in the group.

releaseStack

message

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

libraryStack message, stacksInUse property, start using command, stop using command

Summary

Sent to a stack when the stack is removed from the message path with the stop using command.

Syntax

releaseStack

Example Code

```
on releaseStack
  -- turn off a checkbox to show that this stack is no longer in use
  set the hilite of button (the short name of the target)

  of card "Libraries" to false
end releaseStack
```

Comments

Handle the releaseStack message if you want to perform some task or set a configuration when a stack is no longer in use.

Comments:

The releaseStack message is sent to the stack being released with the stop using command, even if the stack was not in use before the stop using command was executed.

reloadStack

message

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

destroyStack property, mainStacks function, revLoadedStacks function, stacks function

Summary

Sent to a main stack when the user tries to open a main stack with the same name as a previously-opened stack.

Syntax

reloadStack stackName,fileName

Example Code

```
on reloadStack theStack,thePath -- bring reloaded window to front
  -- if you're trying to open the same file (for example, double-
  -- clicking the file), bring the already-open stack to the front:
  if the effective filename of stack theStack is thePath
    then go stack theStack
  else beep -- trying to open a same-named stack in different file
end reloadStack
```

Comments

Handle the reloadStack message when you want to prevent a stack from being reopened, or moderate conflicts between the names of main stacks, or prevent two main stacks with the same name from being open at one time.

Parameters:

The stackName is the short name that belongs to the two conflicting main stacks.

The fileName is the full path to the file that is openingóthe file that contains the second main stack with the same name.

Comments:

The opening of the stack is triggered by the reloadStack message. This means that trapping the reloadStack message and not passing it prevents the second stack from being opened.

If a main stack's `destroyStack` property is set to `false`, the stack remains in memory and can cause a `reloadStack` message to be sent even after its window is closed.

remapColor

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

colorMap property, lockColorMap property, privateColors property, screenColors function

Summary

Has no effect and is included in Transcript for compatibility with imported SuperCard projects.

Syntax

set the remapColor to {true | false}

Example Code

Comments

In SuperCard, the remapColor property determines the behavior of objects when a color table is assigned to a card or stack. In Revolution, color tables are not assigned to objects, and this property has no effect.

The remapColor property is always set to true. A handler can set it to any value without causing a script error, but the actual value is not changed.

remove command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

delete command, group command, groupNames property, place command, About groups and backgrounds, Object menu > Remove Group

Summary

Deletes a group from a specified card.

Syntax

remove group from card

Example Code

```
remove group 1 from this card  
remove group "controls" from recent card  
remove group thisGroup from card "Overview"
```

Comments

Use the remove command to remove a group from a card without removing it from the stack.

Parameters:

The group is a reference to any group that is on the specified card.

The card is any card reference.

Comments:

The group is removed from the card, but is still present in the stack.

remove script

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

backScripts function, frontScripts function, insert script command, script property, stop using command

Summary

Takes an object out of the message path that was placed in the message path with the insert script command.

Syntax

remove [the] script of object from {front | back}

Example Code

```
remove the script of button ID 2 from front  
remove the script of this card from back
```

Comments

Use the remove script command to remove an object from the message path.

Parameters:

The object is any object in an open stack.

Comments:

You can remove a script only if it has previously been inserted in the message path with the insert script command. The remove script command does not affect objects that are normally in the message path or scripts that were placed in the message path with the start using command.

rename

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

create folder command, delete file command, files function, fileType property, revCopyFile command, revCopyFolder command, revMoveFolder command, sysError function, tempName function

Summary

Gives a file or folder a new name or moves it to a new location or both.

Syntax

rename [file | folder |directory] filePath to newPath

Example Code

```
rename file "Foo.txt" to "Bar.txt"  
rename folder "/bin/utilities/help" to "/public/utilities/new-help"
```

Comments

Use the rename command to change the name or location of a file or folder.

Parameters:

The filePath specifies the current name and location of the file or folder you want to rename or move. If you specify a name but not a location, Revolution assumes the file or folder is in the defaultFolder.

The newPath specifies the new name and location of the file or folder.

Comments:

The rename command can be used to change a file's or folder's location only if the old and new locations are both on the same volume.

Caution! This command can be used to rename or move files and folders your stack did not create. Of course, a stack should not rename or move files and folders it didn't create without obtaining explicit confirmation from the user.

Changes to Transcript:

The ability to move a file or folder on Mac OS and Windows systems with the rename command was introduced in version 1.1.1. In previous versions, the rename command could be used only to change the name, not the location.

repeat

control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

each keyword, end repeat keyword, exit repeat control structure, for keyword, forever keyword, next repeat control structure, until keyword, while keyword, Why does a repeat loop behave strangely?, Recipe for listing the unique words in a string, Recipe for reversing a string, Recipe for word-wrapping text to a line length

Summary

Executes a set of statements repeatedly.

Syntax

```
repeat loopForm
  statementList
end repeat
```

Example Code

Comments

Use the repeat control structure to perform the same set of actions for each member of a set: for example, for each card in a stack, or each line in a variable.

Form:

The repeat control structure always begins with the word "repeat".

The last line of a repeat control structure is the end repeat keyword.

Parameters:

The loopForm is one of the following forms:

- ĩ forever
- ĩ until condition
- ĩ while condition
- ĩ [for] number times

```
  i with counter = startValue [to | down to] endValue [step increment]
  i for each chunkType labelVariable in container
  i for each element labelVariable in array
```

The condition is any expression that evaluates to true or false.

The number, startValue, endValue, and increment are numbers or expressions that evaluate to numbers.

The counter or labelVariable is a legal variable name.

The chunkType is one of character (or char), word, line, item, or token.

The container is any existing container. The array is any existing container that contains an array of values.

The statementList consists of one or more Transcript statements, and can also include if, switch, try, or repeat control structures.

Comments:

How many times the statementList is executed depends on the loopForm you use.

The forever form:

The forever form continues repeating the statements in the statementList until an exit, exit repeat, pass, or return statement is executed. Usually, one of these control structures is included in an if control structure within the statementList.

Use the forever form if you want to test a condition at the bottom of the loop, after the statementList is executed. In the following example, the go command is executed at least once, since the mouseClicked is not checked until after the go command:

```
repeat forever
  go next card
  if the mouseClicked then exit repeat -- user clicked
end repeat
```

If no loopForm is specified, the forever form is used.

The until and while forms:

The until condition and while condition forms repeat the statementList as long as the condition is false or as long as it is true, respectively. Revolution re-evaluates the condition before each iteration.

Use the until condition or while condition form if you want to test a condition at the top of the loop, before the statements are executed. This example scrolls through the cards until the user clicks the mouse:

```
repeat until the mouseClicked
  go next
  wait for 100 milliseconds
end repeat
```

The for form:

The for number times form repeats the statementList for the specified number of times.

The number is evaluated when the loop is first entered, and is not re-evaluated as a result of any actions performed in the statementList. For example, if the number is the number of cards, and the statementList contains a create card command, the loop is executed as many times as there were cards when the loop began, even though the current number of cards is changing with each iteration through the loop.

If the number is not an integer, it is rounded to the nearest integer, using the same rules as the round function.

Use the for number times form if you want to execute the statementList a fixed number of times. The following simple example beeps three times:

```
repeat for 3 times
  beep
end repeat
```

The with form:

The with counter = startValue to endValue form and the with counter = startValue down to endValue form set the counter to the startValue at the beginning of the loop, and increase (or decrease, if you're using the down to form) the countVariable by 1 each time through the loop. When the counter is greater than or equal to the endValue, (less than or equal to, if you're using the down to form), the loop performs its final iteration and then ends.

If you specify an increment, the increment is added to the counter each time through the loop, rather than the counter being increased by 1. (The increment is not treated as an absolute value: if you're using the down to form, the increment must be negative.)

As with the for number times form described above, the startValue and endValue are evaluated when the loop is first entered, and are not re-evaluated as a result of any actions performed in the statementList.

Use one of these forms if you want to perform an action on each member of a set, and you need to refer to the member by number within the statementList. The following example loops through all the controls on the current card. The counter x is 1 during the first iteration, 2 during the second, and so on:

```
repeat with x = 1 to the number of controls
  show control x
end repeat
```

The following example loops backwards through a set of lines. The counter myLine is 20 during the first iteration, 18 during the second, and so on:

```
repeat with myLine = 20 down to 1 step -2
  put myLine
end repeat
```


Note: It is possible to change the counter variable in a statement in the loop. However, doing this is not recommended, because it makes the loop logic difficult to follow:

```
repeat with x = 1 to 20 -- this loop actually repeats ten times
  answer x
  add 1 to x -- not recommended
end repeat
```

The for each form:

The for each chunkType labelVariable in container form sets the labelVariable to the first chunk of the specified chunkType in the container at the beginning of the loop, then sets it to the next chunk for each iteration. For example, if the chunkType is word, the labelVariable is set to the next word in the container for each iteration of the loop.

Use the for each form if you want to perform an action on each chunk in a container. This form is much faster than the with countVariable = startValue to endValue form when looping through the chunks of a container. The following example changes a return-delimited list to a comma-delimited list:

```
repeat for each line thisLine in myList
  put thisLine & comma after newList
end repeat
if last char of newList is comma then delete last char of newList
```

The for each element labelVariable in array form sets the labelVariable to the first element in the array at the beginning of the loop, then sets it to the next element for each iteration.

Important! You cannot change the labelVariable in a statement inside the loop. Doing so will cause a script error. You can change the content of the container, but doing so will probably produce unexpected results.

Use the for each form if you want to perform an action on each element in an array. The following example gets only the multi-word entries in an array of phrases:

```
repeat for each element thisIndexTerm in listOfTerms
  if the number of words in thisIndexTerm > 1
  then put thisIndexTerm & return after multiWordTerms
end repeat
```

Note: The repeat control structure is implemented internally as a command and appears in the commandNames.

Changes to Transcript:

The ability to specify an increment for the repeat with counter = startValue to endValue form was added in version 2.0. In previous versions, this form of the repeat control structure always incremented or decremented the counter by 1 each time through the loop.

repeatCount

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentFrame property, frameCount property, looping property, palindromeFrames property

Summary

Specifies the number of times an animated GIF image repeats.

Syntax

set the repeatCount of image to numberOfCycles

Example Code

```
set the repeatCount of image 1 to 0 -- stop looping
```

Comments

Use the repeatCount property to specify how many times an animated GIF image should repeat when it's displayed, or to stop it from repeating.

Value:

The repeatCount of an image is an integer.

Comments:

If the repeatCount is positive, the image repeats the specified number of times. If the repeatCount is negative, the image continues repeating indefinitely while it is being displayed.

If the repeatCount is zero, the image does not repeat at all. If you set the repeatCount of an animated GIF image to zero while it is playing, it stops repeating immediately.

If the contents of the image is not an animated GIF, the setting of the repeatCount property has no effect.

repeatDelay

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

mouseDown message, mouseStillDown message

Summary

Specifies how long a scrollbar or field waits between the click in the scrollbar and the time the scrollbar movement begins to repeat.

Syntax

set the repeatDelay to milliseconds

Example Code

```
set the repeatDelay to 10
```

Comments

Use the repeatDelay property to control the responsiveness of scrollbars.

Value:

The repeatDelay is a non-negative integer.

By default, the repeatDelay property is set to 5 milliseconds.

Comments:

When the user clicks in a scrollbar, a mouseDown message is sent to the scrollbar. If the mouse button is still being held down after the time specified by the repeatDelay property, the scrollbar's movement begins repeating.

repeatRate

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

idleRate property, scrollbarLineDec message, scrollbarLineInc message, scrollbarPageDec message, scrollbarPageInc message

Summary

Specifies how long a scrollbar waits between repeats when the mouse button is held down.

Syntax

set the repeatRate to milliseconds

Example Code

```
set the repeatRate to 1000 -- 1 second
```

Comments

Use the repeatRate property to control auto-scrolling when the mouse button is held down.

Value:

The repeatRate is an integer.

By default, the repeatRate property is set to 50 (1/20th of a second).

Comments:

If the user clicks in the gray region of a scrollbar or on one of its arrows and holds the mouse button down, the scroll action is repeated and the scrollbar is updated repeatedly as long as the mouse is down. The repeatRate is the time in milliseconds between one update and the next.

If the repeatRate is zero, updates are performed as often as possible with no delay.

If the repeatRate is negative, scrollbar clicks do not repeat at all.

replace

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

caseSensitive property, find command, matchText function, offset function, replaceText function, Recipe for removing boldfacing from a field

Summary

Replaces text in a container with other text.

Syntax

replace oldString with newString in container

Example Code

```
replace "A" with "N" in thisVariable -- changes all As to Ns
replace return with empty in field 1 -- runs lines together
replace somethingOld with " in it
replace ".com" with somethingNew in URL "file:stuff.txt"
```

Comments

Use the replace command to replace all instances of one string with another string.

Parameters:

The oldString is any expression that evaluates to a string, and specifies the text to replace.

The newString is any expression that evaluates to a string, and specifies the text to replace the oldString with.

The container is a field, button, or variable, or the message box.

Comments:

The replace command is faster than the replaceText function, but does not support regular expressions: you can replace only an exact string of text.

Important! You can use the replace command on a field, but doing so removes any formatting (fonts, styles, colors, and sizes) in the field. To work around this limitation, use the field's htmlText property as the source for replacement instead of using the field itself as the source:

get the htmlText of field "Stuff"
replace "old" with "new" in it
set the htmlText of field "Stuff" to it

replaceText

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

caseSensitive property, filter command, matchText function, offset function, replace command, Regular Expressions Syntax Reference

Summary

Searches for a regular expression and replaces the portions that match the regular expression.

Syntax

`replaceText(stringToChange,matchExpression,replacementString)`

Example Code

```
replaceText("malformed","mal","well") -- returns "wellformed"  
replaceText(field "Stats",return,comma) -- makes comma-delimited
```

Comments

Use the replaceText function to search for and replace text that matches a particular pattern.

Parameters:

The stringToChange is any expression that evaluates to a string.

The matchExpression is a regular expression.

The replacementString is any expression that evaluates to a string.

Value:

The replaceText function returns the changed string.

Comments:

The replaceText function replaces all the occurrences of the matchExpression with the replacementString. If more than one matching substring is found, the replaceText function replaces all of them.

The `replaceText` function is not as fast as the `replace` command, but is more flexible because you can search for any text that matches a regular expression.

The `stringToChange` and `matchExpression` are always case-sensitive, regardless of the setting of the `caseSensitive` property. (If you need to make a case-insensitive comparison, use "(?i)" at the start of the `matchExpression` to make the match case-insensitive.)

reply command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

address property, appleEvent message, request appleEvent command, request command, send to program command

Summary

Returns data to an application that sent Revolution an Apple event.

Syntax

reply string [with keyword aeKeyword]

reply error string

Example Code

```
reply "Connection established"
reply line thisLine of field "AE Replies"
reply "45" with keyword "errn"
reply error "Not found."
```

Comments

Use the reply command to interact with another application via Apple events.

Parameters:

The string is the string to send as Apple event data.

The aeKeyword is the Apple event keyword.

Comments:

The reply command for inter-application communication is analogous to the return control structure for inter-handler communication.

When a program responds to an Apple event, it returns several different pieces of information. Each piece corresponds to an Apple event keyword. Use the reply with aeKeyword form of this command to specify which pieces you want to reply with.

The form

reply string
is equivalent to
reply string with keyword "----"

The form
reply error string
is equivalent to
reply string with keyword "errs"

For more information about Apple events, see Apple Computer's technical documentation, Inside Macintosh: Interapplication Communication, located at
<<http://developer.apple.com/techpubs/mac/IAC/IAC-2.html>>.

request command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

address property, appleEvent message, reply command, request appleEvent command, send to program command

Summary

The request data gets information from another application via Apple events.

Syntax

request expression {of | from} {program | application} programAddress

Example Code

```
request "the hilite of button 3" from program "HyperCard"
```

```
request field "Appname" from program "Ozone:Geek Paradise:FileMaker"
```

Comments

Use the request command to obtain data from another application via the eval Apple event.

Parameters:

The expression is the string you're asking the other application to evaluate. Its exact format and meaning depends on the other application.

The programAddress is the AppleTalk address of the other program. The AppleTalk address consists of three parts, separated by colons: the zone the other computer is in, the name of the computer, and the name of the target program. If the other computer is in the same zone as yours, you can omit the zone. If the other program is running on the same computer, you can omit both the zone and the computer name.

Comments:

The request command sends an eval Apple event to the programAddress. If the program supports this Apple event, it evaluates the expression and sends the result back to Revolution, which places it in the it variable.

If the program sends back an error message, it is placed in the result function.

For more information about Apple events, see Apple Computer's technical documentation, Inside Macintosh: Interapplication Communication, located at
<<http://developer.apple.com/techpubs/mac/IAC/IAC-2.html>>.

request appleEvent

command

Synonyms

request ae

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

address property, appleEvent message, reply command, request command, send to program command

Summary

Gets data about an Apple event that was sent to Revolution.

Syntax

```
request appleEvent {class|ID|sender|returnID|data [with keyword aeKey]}
```

Example Code

```
request appleEvent class
request appleEvent data
request appleEvent data with keyword "timo"
```

Comments

Use the request appleEvent command from within an appleEvent handler to extract information about a pending Apple event.

Parameters:

The aeKey is an Apple event keyword. If you do not specify an aeKey (or any of the specific forms of the command), the request appleEvent command gets the data attached to the Apple event.

Comments:

The information returned by the request appleEvent command is placed in the it variable.

When a program sends an Apple event, it includes several different pieces of information. Each piece corresponds to an Apple event keyword. Use the request appleEvent data with aeKey form of this command to specify which pieces you want to get.

ï The form request appleEvent class is equivalent to request appleEvent data with keyword "evcl". It gets the four-character class the Apple event belongs to (for example, misc or aevt).

- ï The form request appleEvent ID is equivalent to request appleEvent data with keyword "evid". It gets the four-character name of the specific Apple event (for example, dosc or eval).
- ï The form request appleEvent sender is equivalent to request appleEvent data with keyword "addr". It gets the AppleTalk address of the program that sent the Apple event to Revolution.
- ï The form request appleEvent returnID is equivalent to request appleEvent data with keyword "rtid". It gets the Apple event's return ID number.
- ï The form request appleEvent data gets the data associated with the Apple event. The meaning of this data depends on which kind of Apple event is being handled. For example, the odoc (open document) event sends the file path of the document being opened as the Apple event data.

For more information about Apple events, see Apple Computer's technical documentation, Inside Macintosh: Interapplication Communication, located at
<<http://developer.apple.com/techpubs/mac/IAC/IAC-2.html>>.

reset
command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

reset cursors command, reset paint command, templateAudioclip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoclip keyword

Summary

Returns the specified template object to its default settings.

Syntax

reset [the] template {objectType}

Example Code

```
reset the templateGraphic  
reset the templateStack
```

Comments

Use the reset command to change a template object back to its defaults after you have changed it temporarily.

Parameters:

The objectType is one of Revolution's objects: stack, group, card, field, button, graphic, image, scrollbar, player, audioClip, videoClip, or EPS.

Comments:

Change the properties of template objects temporarily to easily create a number of objects of that type with the property settings you want.

Note: The Revolution development environment makes changes to template objects when it creates new objects. This means that objects you create in the development environment may not have the same property settings as objects you create in a standalone application.

reset cursors

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

choose command, cursor property, reset command, reset paint command

Summary

Reloads the standard set of cursors.

Syntax

reset cursors

Example Code

```
reset cursors
```

Comments

Use the reset cursors command to change the set of cursors used by Revolution without restarting the application.

Comments:

Revolution's built-in cursors are used whenever the cursor property has not been set to a custom cursor. If you change the set of built-in cursors, you must either quit and restart the application or use the reset cursors command to begin using the new cursor shapes.

You can force Revolution to use the system cursors with the following statements:

```
delete stack "revCursors"  
reset cursors
```

Caution! If you use the delete stack command to remove the "revCursors" stack, Revolution's cursors are permanently deleted and you will need to download a new cursors stack to restore them.

reset paint command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

reset command, reset cursors command, revert command

Summary

Restores the default settings of the global properties that apply to painting.

Syntax

reset paint

Example Code

```
reset paint  
if the mouse is down then reset paint
```

Comments

Use the reset paint command as a shortcut to clean up paint properties by resetting them to their default settings.

Comments:

The reset paint command assigns the following property values:

- The brush property is set to 8.
- The brushColor property is set to white.
- The centered property is set to false.
- The eraser property is set to 2.
- The filled property is set to false.
- The grid property is set to false.
- The gridSize property is set to 8.
- The lineSize property is set to 1.
- The pattern property is set to 1.
- The penColor property is set to black.
- The polySides property is set to 4.
- The roundEnds property is set to false.
- The slices property is set to 16.

i The spray property is set to 31.

resetAll

command

Synonyms

libURLResetAll

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close socket command, ftp keyword, http keyword, openSockets function, reset command, socketError message, How to unwedge upload and download operations

Summary

Closes all open sockets and halts all pending Internet operations.

Syntax

resetAll

Example Code

```
if the openSockets is not empty then resetAll
```

Comments

Use the resetAll command to troubleshoot Internet connection problems.

Comments:

The resetAll command closes all open sockets, including any sockets that have been opened by the Internet library; clears all variables used by the Internet library; and clears all cached data.

Since the Internet library's routines for handling http and ftp uploads and downloads are socket-based, the resetAll command resets all these operations.

Caution! The resetAll command closes all open sockets, which includes any other sockets opened by your application and any sockets in use for other uploads and downloads. Because of this, you should avoid routine use of the resetAll command. Consider using it only during development, to clear up connection problems during debugging.

Important! The resetAll command is part of the Internet library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Internet library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `resetAll` command is not yet available. This may affect attempts to use this command in `startup`, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `resetAll` command can be used in any handler.

resfile

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

binfile keyword, copyResource function, deleteResource function, file keyword, getResource function, getResources function, http keyword, URL keyword, About filename specifications and file paths, About using URLs, uploading, and downloading, How to associate files with a Mac OS standalone application, How to associate files with an OS X standalone application, How to associate files with a Windows standalone application, How to copy a resource fork

Summary

Used as a URL type with such commands as put and get to designate the resource fork of a Mac OS file.

Syntax

Example Code

```
get URL "resfile:/Disk/System Folder/Apple Menu Items/Calculator App"
```

Comments

Use the resfile keyword to work with Mac OS resource forks.

Comments:

On Mac OS systems, files consist of either or both of a data fork and a resource fork. The resource fork contains defined resources such as icons, menu definitions, dialog boxes, fonts, and so forth.

A URL container can be used anywhere another container type is used. The Transcript-specific scheme "resfile" indicates the resource fork of a file which is located on the user's system. The file is specified by either an absolute path starting with "/", or a relative path from the defaultFolder.

A resfile URL specifies the entire resource fork, not just one resource. The most common use for this URL scheme is to copy an entire resource fork from one file to another. To modify the data from a resfile URL, you need to understand the details of Apple's resource fork format.

Important! Unlike the file and binfile URL types, the resfile keyword cannot be used to create a file. If the file "myFile" doesn't yet exist, attempting to use it with the resfile keyword will fail and the result

function will be set to "file not found". To create a new resource file, first use a file URL to create the file with an empty data fork, then write the needed data to its resource fork:

```
put empty into URL "file:myFile" -- creates an empty file  
put myStoredResources into URL "resfile:myFile"
```

resizable

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

decorations property, maxHeight property, maxWidth property, minHeight property, minWidth property, resizeStack message, How to prevent the user from resizing a window

Summary

Specifies whether the user can resize a stack window.

Syntax

set the resizable of stack to {true | false}

Example Code

```
set the resizable of this stack to false
```

Comments

Use the resizable property to control whether the user can change a window's size.

Value:

The resizable of a stack is true or false.

By default, the resizable property of newly created stacks is set to true.

Value:

The resizable property controls only whether the user can change the stack's size. Even if the resizable is false, you can use the rectangle property (and related properties) to change the stack's size in a handler.

If the stack's style is "modal", or if it has been opened with the modal command, the setting of the resizable property has no effect, and the stack window cannot be resized.

resizeControl

message

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

moveControl message, rectangle property, resizeStack message

Summary

Sent to a control right after the user resizes it.

Syntax

resizeControl

Example Code

```
on resizeControl -- make sure a label stays at the bottom edge
  set the top of field "Label" to the bottom of me
  pass resizeControl
end resizeControl
```

Comments

Handle the resizeControl message if you want to update other objects, or do other tasks, in response to the user resizing a control.

Comments:

The resizeControl message is only sent when the user resizes a control by dragging its handles. It is not sent if a handler changes the size of a control by changing its properties (width, height, and so on).

The resizeControl message is sent after the resizing is finished. This means that you cannot prevent a control's size from being changed by trapping this message.

resizeStack

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

height property, iconifyStack message, liveResizing property, moveStack message, rectangle property, resizable property, resizeControl message, revChangeWindowSize command, unIconifyStack message, width property, How to automatically adjust objects when a window is resized, How to respond to resizing a window

Summary

Sent to the current card when the stack window is resized.

Syntax

resizeStack newWidth,newHeight,oldWidth,oldHeight

Example Code

```
on resizeStack newWidth,newHeight -- put an object in the middle
    set the location of graphic "Middle"

        to newWidth div 2,newHeight div 2
        pass resizeStack
    end resizeStack
```

Comments

Handle the resizeStack message if you want to update the position of objects or do other tasks when the stack window changes size.

Parameters:

The newWidth is the stack's new width in pixels.

The newHeight is the stack's new height in pixels.

The oldWidth is the stack's original width in pixels.

The oldHeight is the stack's original height in pixels.

Comments:

The `resizeStack` message is sent when the user resizes the stack by dragging its size box. It is also sent if a handler changes the size of the stack by changing its properties (width, height, and so on).

The `resizeStack` message is sent after the resizing is finished. This means that you cannot prevent a stack's size from being changed by trapping this message. If the stack's `liveResizing` property is true, `resizeStack` messages are sent continuously during resizing, but you still cannot prevent resizing by trapping the message.

If the `liveResizing` property is true, the `oldWidth` and `oldHeight` for each `resizeStack` message is the same as the `newWidth` and `newHeight` for the previous `resizeStack`. The stack's original width and height are passed only with the first `resizeStack` message sent during a resize operation.

The screen is locked while a `resizeStack` handler is running, so it is not necessary to use the lock screen command to prevent changes from being seen. (However, the `lockScreen` property is not set to true.)

Note: If the stack's `vScroll` property is nonzero, the amount of scroll is not included in the `newHeight` and `oldHeight`. This means that the parameters of the `resizeStack` message are always equal to the stack's height before and after resizing, regardless of the `vScroll` setting.

result function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

on control structure, return control structure, sysError function

Summary

Returns the status of the last command that was executed.

Syntax

the result
result()

Example Code

```
the result  
if the result is not empty then exit mouseUp
```

Comments

Use the result function to check whether the last command executed encountered an error.

Value:

The result function returns a string.

Comments:

Many commands (such as go and find) set the value of the result function when they finish. In most cases, if the result is empty, the command was successful; if the result is not empty, the command failed and the result contains an error message. See the specific command for information about whether it sets the result function.

If a command fails because of an operating-system error (such as a file not being available), the sysError function returns the error message that the operating system reported to Revolution. In this case, you can use the sysError function to get additional information about the error if the result is not empty.

If the return control structure appears within an on handler, the result function is set to the return value. If a handler contains the lines
myMessage

put the result
and the "myMessage" handler contains the line
return "some value"

the text "some value" appears in the message box when the handler runs. You can use this capability to return an error message if a custom message handler fails.

The result function is set to empty when the current handler finishes executing.

Important! If you need to check the result, do so immediately after the statement whose result you want to check (or save the value of the result in a variable for later checking). This ensures that the result you are checking is the correct one, not a value set by a later statement.

resume
message

Synonyms

Objects
card

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
focusIn message, openStack message, startup message, resumeStack message, suspend message

Summary
Sent to the current card when the application is brought to the foreground.

Syntax
resume

Example Code

```
on resume
  show stack "Toolbar"
end resume
```

Comments
Handle the resume message if you want to perform some action when the application is made active.

Comments:
The resume message is sent whenever the user switches back to the application from another program.

The actual switch is not triggered by the resume message, so trapping the message and not allowing it to pass does not prevent the user from switching into the application.

resumeStack

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

focusIn message, openStack message, resume message, suspendStack message, unIconifyStack message

Summary

Sent to the current card when a stack window is brought to the front.

Syntax

resumeStack

Example Code

```
on resumeStack -- show a palette that only applies to this window
  show stack "Accessories"
end resumeStack
```

Comments

Handle the resumeStack message if you want to perform some action when a stack window is made active.

Comments:

The resumeStack message is sent whenever a stack window becomes the active window.

The actual window activation process is not triggered by the resumeStack message, so trapping the message and not allowing it to pass does not prevent the stack window from becoming the active window.

retainImage

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

postScript property, retainPostScript property

Summary

Specifies whether an EPS object contains a preview bitmap.

Syntax

set the retainImage of EPSObject to {true | false}

Example Code

```
set the retainImage of EPS 3 to false
```

Comments

Use the retainImage property to create a preview bitmap that can be seen on systems without Display PostScript.

Value:

The retainImage of an EPS object is true or false.

By default, the retainImage property of newly created EPS objects is set to false.

Comments:

If an EPS object's retainImage property is true, the non-PostScript preview image attached to the Encapsulated PostScript file is kept with the EPS object and used for display on systems that do not support Display PostScript.

If the retainImage property is false, the preview image is deleted.

This property is supported only on Unix systems with Display PostScript installed.

retainPostScript

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

postScript property, retainImage property

Summary

Specifies whether an EPS object contains the PostScript code.

Syntax

set the retainPostScript of EPSObject to {true | false}

Example Code

```
set the retainPostScript of the EPSTemplate to true
```

Comments

Use the retainPostScript property to keep or ignore the underlying PostScript code of an EPS object after rendering it.

Value:

The retainPostScript of an EPS object is true or false.

By default, the retainPostScript property of newly created EPS objects is set to true.

Comments:

If an EPS object's retainPostScript property is true, the PostScript code used to draw the object is kept as the object's postScript property. This means the EPS object can be printed or scaled and re-drawn without loss of quality.

If the retainPostScript property is false, the PostScript code is deleted once the object has been drawn.

This property is supported only on Unix systems with Display PostScript installed.

return

constant

Synonyms

CR,linefeed,LF

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command, CRLF constant, lineDelimiter property, read from file command, write to file command, Why is there a problem with line endings?, Recipe for collecting text selections on the clipboard

Summary

Equivalent to the line feed character (ASCII 10, Control-J).

Syntax

Example Code

```
put return after word 2 of theData  
repeat until thisChar is return
```

Comments

Use the return constant as an easier-to-read substitute for numToChar(10).

Comments:

The return constant is needed because you can't type the character it represents in a script.

The return constant is a synonym for linefeed. This is different from some other languages, in which return is equivalent to the carriage return character (ASCII 13, Control-M). For most purposes, Revolution translates the linefeed constant and its synonyms into the appropriate end-of-line character for the current operating system. However, you should keep this nuance in mind when processing data from another system, which Revolution has not translated: return is not ASCII 13.

The line feed character is the standard end-of-line delimiter on Unix systems. The end-of-line delimiter for Mac OS systems is a carriage return, and the end-of-line delimiter for Windows systems is a carriage return followed by a line feed. Internally, Revolution always uses a line feed to end lines.

Note: If you specify text mode with the open driver, open file, or open process commands, Revolution translates line feed characters to the appropriate end-of-line marker for the current platform before

writing data, and translates the current platform's end-of-line delimiter to a line feed after reading data. If you specify binary mode with these commands, Revolution does not perform this automatic translation. Likewise, if you put data into a file URL or get data from it, end-of-line translation is performed, but not if you put data into or get data from a binfile URL.

Changes to Transcript:

The LF synonym was added in version 2.0.

return

control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

exit control structure, function control structure, getProp control structure, merge function, on control structure, pass control structure, result function, setProp control structure, throw control structure, How to return an array from a function, How to return multiple values from a handler, Recipe for collecting text selections on the clipboard

Summary

Stops the current handler and returns a value to the handler that called the current handler.

Syntax

return value

Example Code

Comments

Use the return control structure to return a value from a custom function or getProp handler, or to return an error message from a message handler or setProp handler.

Form:

The return statement appears on a line by itself, anywhere inside a handler.

Parameters:

The handler is the name of the handler in which the return control structure appears.

Comments:

When the return control structure is executed, any remaining statements in the handler are skipped. Hence, the return control structure is usually used either at the end of a handler or within an if control structure.

If the return control structure is within a function or getProp control structure, the value is returned to the calling handler as the function value or property setting. For example, if you have the following function handler:

```
function simpleFunction
  return 1
end simpleFunction
```

which is called in the following statement:

```
put simpleFunction() into field 1
```

then 1, the value returned by "simpleFunction", is placed in the field.

If the return control structure is within an on or setProp control structure, the value can be retrieved by checking the result function in the calling handler. Usually, when the return control structure is used within an on or setProp control structure, it returns an error message. (If you want a handler to compute a value as its main reason for existence, you should implement it as a custom function rather than a custom command.)

When a message handler executes a return statement, the message stops and is not passed to the next object in the message path. To halt the current message handler and pass the message on through the message path, use the pass control structure instead. To halt the current handler without returning a result, use the exit control structure instead.

Note: The return control structure is implemented internally as a command and appears in the commandNames.

returnInField

message

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

enterInField message, returnKey message, Recipe for a Find field

Summary

Sent to a field when the selection is in the field and the user presses the Return key.

Syntax

returnInField

Example Code

```
on returnInField -- automatically re-sort a field when a line is added
    sort lines of the target
end returnInField
```

Comments

Handle the returnInField message when you want to perform an action (such as adding a column of figures) when the user presses Return in a field.

Comments:

The Return key (confusingly labeled "Enter" on some keyboards) is usually located above the right-hand Shift key. It is the key you press to go to a new line.

If the returnInField handler does not pass the message or send it to a further object in the message path, the keypress has no effect. Passing the message allows the keypress to have its normal effect.

The returnInField message is sent to buttons whose menuMode is "comboBox", since the type-in box in a combo box behaves like a field.

If there is no selection or insertion point in any field and the user presses Return, the returnKey message is sent instead of returnInField.

returnKey

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

enterKey message, returnInField message, tabKey message, How to make the Return or Enter key activate a button

Summary

Sent to the active object when there is no text selection and the user presses the Return key.

Syntax

returnKey

Example Code

```
on returnKey -- go to the next card when the user presses return
    go next card
end returnKey
```

Comments

Handle the returnKey message when you want to perform an action when the user presses Return.

Comments:

The Return key (confusingly labeled "Enter" on some keyboards) is usually located above the right-hand Shift key. It is the key you press to go to a new line.

The message is sent to the active (focused) control, or to the current card if no control is focused. If there is a text selection or insertion point in a field (or a button whose menuMode is "comboBox") and the user presses Return, the returnInField message is sent instead of returnKey.

revAddXMLNode

command

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revAppendXML command, revDeleteXMLNode command, revEndXMLNode message, revPutIntoXMLNode command, revStartXMLNode message, revSetXMLAttribute command

Summary

Adds a child node to the specified node in an XML tree.

Syntax

revAddXMLNode treeID,parentPath,nodeName,nodeContents

Example Code

```
revAddXMLNode 9, "/", "Balls", ""  
revAddXMLNode myTree, theNode, the short name of me, field "Contents"
```

Comments

Use the revAddXMLNode command to add a node to an XML tree.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The parentPath is the path of the node whose child the node being created will be.

The nodeName is the name of the new node.

The nodeContents is the text to place in the new node.

Comments:

If the revAddXMLNode command encounters an error, the result is set to an error message beginning with "xmlerr".

Important! The revAddXMLNode command is part of the XML library. To ensure that the command works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revAppendXML

command

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revAddXMLNode command, revCreateXMLTree function, revCreateXMLTreeFromFile function, revPutIntoXMLNode command, revSetXMLAttribute command, revXMLAddDTD command, revXMLText function

Summary

Adds XML text to an XML tree.

Syntax

revAppendXML treeID,parentNode,newXML

Example Code

```
revAppendXML 4,"/Publications/Fiction/SF",URL "file:SF Books.xml"  
revAppendXML thisTree,the currNode of me,the text of me
```

Comments

Use the revAppendXML command to add new XML data to an existing XML tree.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The parentNode is the path to the node where the newXML will be inserted. Any elements in the newXML will become child nodes of the parentNode. If no parentNode is specified, the newXML is inserted at the end of the XML tree.

The newXML is the XML text to be inserted.

Comments:

If the revAppendXML command encounters an error, the result is set to an error message beginning with "xmlerr".

Important! The revAppendXML command is part of the XML library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revAppVersion

function

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

buildNumber function, libURLVersion function, version function, How to investigate the Revolution custom libraries

Summary

Returns the version of the Revolution application.

Syntax

revAppVersion()

Example Code

```
revAppVersion()  
if revAppVersion() is not "1.0" then answer "Sorry, Charlie!"
```

Comments

Use the revAppVersion function if you need to check which version of the development environment is currently running.

Value:

The revAppVersion function returns a string indicating the version of the development environment.

Comments:

The revAppVersion function is different from the version function. The revAppVersion function returns the version of the development environment, while the version function returns the version of the underlying engine. For example, the contents of the Revolution menu bar, the Revolution custom libraries, and the property inspector depend on the revAppVersion, but the features of the Transcript language depend on the engine version.

Important! The revAppVersion function is part of the Common library. To ensure that the function works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The revAppVersion function is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revAppVersion` function is not yet available. This may affect attempts to use this function in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revAppVersion` function can be used in any handler.

revCacheGeometry

command

Synonyms

Objects

Geometry library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

lock messages command, moveControl message, resizeControl message, revUpdateGeometry command, Geometry Management Tutorial, How to investigate the Revolution custom libraries, Why does the Geometry pane move objects to the wrong place?

Summary

Updates Geometry settings.

Syntax

revCacheGeometry

Example Code

```
revCacheGeometry  
if the rect of me is not savedRect then revCacheGeometry
```

Comments

Use the revCacheGeometry command to update the Geometry pane's internal settings, after changing the size or position of controls that you have used the Geometry pane to specify behavior for.

Comments:

The revCacheGeometry command rebuilds the Geometry pane's internal settings for the controls on the current card, storing the baseline position that the automatic positioning and scaling uses as a starting point.

The Revolution development environment automatically executes the revCacheGeometry command when a resizeControl or moveControl message is sent. This means that when you move or resize a control in the development environment, its stored Geometry baseline is normally updated automatically.

Normally, you do not need to use the revCacheGeometry command at all, since Revolution automatically updates the geometry settings when a control is moved or resized. However, if you move or resize controls manually or in a handler, and the resizeControl or moveControl message is not sent, you may need to use the revCacheGeometry command to update the baseline positions.

To update the settings, use the following statement in a handler or the message box, before changing cards or resizing the window:

```
revCacheGeometry
```

Important! The `revCacheGeometry` command is part of the Geometry library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Geometry Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Geometry library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revCacheGeometry` command is not yet available. This may affect attempts to use this command in `startup`, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revCacheGeometry` command can be used in any handler.

revChangeWindowSize

command

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

formattedHeight property, formattedWidth property, height property, location property, lockLocation property, rectangle property, resizeStack message, width property, How to change the size and position of a window, How to investigate the Revolution custom libraries

Summary

Changes the height and width of a stack window.

Syntax

```
revChangeWindowSize newWidth,newHeight[,anim[,cardNumber[,totalTime]]]
```

Example Code

```
revChangeWindowSize 100,120
revChangeWindowSize the width of stack "Template",200,"slide"
revChangeWindowSize myWidth,myHeight,"snap",1 -- displays card 1
revChangeWindowSize 200,250,"slide",,500 -- slides over 1/2 second
```

Comments

Use the revChangeWindowSize command to change the size of a stack while leaving its top left corner in place, with optional animation effects.

Parameters:

The newWidth is a positive integer.

The newHeight is a positive integer.

The anim is either "snap" or "slide". If the anim is "slide", the height and width are changed during the length of time specified in totalTime, with the window edges sliding gradually instead of snapping to their new position. If the anim is "snap", or if no anim is specified, the height and width are changed immediately.

The cardNumber is the number of a card in the stack. The stack displays the card with this number during the transition, and returns to the original card at the end. If the anim is not "slide", the cardNumber has no effect.

The `totalTime` is the number of milliseconds to take for the slide effect. If the `anim` is not "slide", the `totalTime` has no effect and the change takes place immediately. By default, the `totalTime` is 250 (1/4 second).

Comments:

The `revChangeWindowSize` command resizes a stack window to the `newWidth` and `newHeight`.

There are two differences between using the `revChangeWindowSize` command and simply changing the stack's height and width properties:

- The `revChangeWindowSize` command shrinks or grows the window from its top left corner: the window's top and left edges stay in place, while the bottom and right edges move. Setting the height and width properties leaves the center of the window in place while all four edges move inward or outward.
- The `revChangeWindowSize` command lets you change the window size by optionally sliding the edges over a specified time period, and displaying a card during the transition.

Important! The `revChangeWindowSize` command is part of the Common library. To ensure that the command works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The `revChangeWindowSize` command is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revChangeWindowSize` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revChangeWindowSize` command can be used in any handler.

revCloseCursor

command

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revDatabaseCursors function, revCloseDatabase command, About connecting to and using SQL databases, How to create a record set in a SQL database, How to find out which record sets are open in a database

Summary

Closes a record set (database cursor).

Syntax

revCloseCursor recordSetID

Example Code

```
revCloseCursor 3  
revCloseCursor finalResults
```

Comments

Use the revCloseCursor command to clean up after using a record set in a database.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Comments:

All open record sets are closed automatically when the database they are associated with is closed. However, it is recommended practice to explicitly close record sets when your stack is finished with them.

If the closure is not successful, the result is set to an error message.

Important! The revCloseCursor command is part of the Database library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revCloseDatabase

command

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCloseCursor command, revOpenDatabase function, revOpenDatabases function, About connecting to and using SQL databases, How to close the connection to a database

Summary

Closes the connection to a database.

Syntax

revCloseDatabase databaseID

Example Code

```
revCloseDatabase savedBudgetDB
```

Comments

Use the revCloseDatabase command to cleanly log off from a database.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

Comments:

All open databases are closed automatically when the application quits. However, it is recommended practice to explicitly close database connections when your stack is finished with them.

The revCloseDatabase command closes all open record sets (database cursors) opened by the database, as well as closing the database itself.

If the closure is not successful, the result function returns an error message that begins with the string "revdberr".

Important! The revCloseDatabase command is part of the Database library. To ensure that the command works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revCloseVideoGrabber

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

close command, revInitializeVideoGrabber command, revStopPreviewingVideo command, revStopRecordingVideo command

Summary

Closes the video grabber window.

Syntax

revCloseVideoGrabber

Example Code

```
revCloseVideoGrabber  
if there is no stack "Video Control" then revCloseVideoGrabber
```

Comments

Use the revCloseVideoGrabber command to free up memory when you're done using the Video library.

Comments:

The Video library loads the operating system's video capture software into memory when you use the revInitializeVideoGrabber command. The revCloseVideoGrabber command unloads this software, freeing up the memory it uses, when you're done.

If your application uses video capture, you should execute the revCloseVideoGrabber command either when your application is finished using video capture, when the stack that uses video capture is closed (in a closeStack handler), or when your application quits (in a shutdown handler).

If the video grabber was already recording video to a file, the revCloseVideoGrabber command stops the recording.

Important! The revCloseVideoGrabber command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revCommitDatabase

command

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revQueryResult function, revRollBackDatabase command, About connecting to and using SQL databases, How to save changes to a SQL database

Summary

Saves recent changes to a database.

Syntax

revCommitDatabase databaseID

Example Code

```
revCommitDatabase 4  
revCommitDatabase currentDB
```

Comments

Use the revCommitDatabase command to save the database.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

Comments:

If the operation is not successful, the revCommitDatabase command returns an error message that begins with the string "revdberr".

Note: Not all databases support the revCommitDatabase command.

Important! The revCommitDatabase command is part of the Database library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revCopyFile

command

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer file command, binfile keyword, create alias command, delete file command, file keyword, files function, resfile keyword, revCopyFolder command, rename command, About filename specifications and file paths, How to investigate the Revolution custom libraries, Why can't Revolution find a file I specified?

Summary

Copies a file.

Syntax

revCopyFile fileToCopy,folderToCopyTo

Example Code

```
revCopyFile "/Disk/myfile", "/Disk/Folder/"  
revCopyFile "data/settings.txt", "olddata"  
revCopyFile the sourceFile of this card, it
```

Comments

Use the revCopyFile command to make a copy of a file to another folder.

Parameters:

The fileToCopy specifies the name and location of the file. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

The folderToCopyTo specifies the name and location of the folder where the copied file should be placed. If you specify a name but not a location, Revolution assumes the destination folder is in the defaultFolder.

Comments:

The revCopyFile command uses system services on each platform to perform the copy. On Mac OS systems, it uses AppleScript; on OS X, Windows and Unix systems, it uses the shell function. Any errors encountered are returned in the result function.

Important! To copy a bundle on OS X systems, you must use the `revCopyFolder` command instead.

You can also copy a file using the `put` command, in a statement like the following:

```
put URL "binfile:/Disk/myfile" into URL "binfile:/Disk/Folder/myfile"
```

However, the `revCopyFile` command provides certain advantages. It copies file attributes (such as file type) and Mac OS resource forks along with the file. It also does not require reading the entire file into memory, so even extremely large files can be copied.

Important! The `revCopyFile` command is part of the Common library. To ensure that the command works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The `revCopyFile` command is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revCopyFile` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revCopyFile` command can be used in any handler.

revCopyFolder

command

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer folder command, create alias command, create folder command, folders function, revCopyFile command, revMoveFolder command, About filename specifications and file paths, How to investigate the Revolution custom libraries

Summary

Copies a folder with all its contents.

Syntax

revCopyFolder folderToCopy,destinationFolder

Example Code

```
revCopyFolder "E:/Settings","C:/Program Files/My App/Settings"  
revCopyFolder "data","backups" -- creates "data/backups/"
```

Comments

Use the revCopyFolder command to make a copy of a folder inside another folder.

Parameters:

The folderToCopy specifies the name and location of the folder. If you specify a name but not a location, Revolution assumes the folder is in the defaultFolder.

The destinationFolder specifies the name and location of the folder where the copy will be created. If you specify a name but not a location, Revolution assumes the destination folder is in the defaultFolder.

Comments:

The revCopyFolder command makes a copy of the entire folder, including all files, subfolders, and their contents. The folder remains in its original location and the copy is placed in the new location.

The revCopyFolder command uses system services on each platform to perform the copy. On Mac OS and OS X systems, it uses AppleScript; on Windows and Unix systems, it uses the shell function. Any errors encountered are returned in the result function.

Important! The revCopyFolder command is part of the Common library. To ensure that the command works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The revCopyFolder command is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the revCopyFolder command is not yet available. This may affect attempts to use this command in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the revCopyFolder command can be used in any handler.

revCreateXMLTree

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCreateXMLTreeFromFile function, revDeleteAllXMLTrees command, revDeleteXMLTree command, revEndXMLNode message, revEndXMLTree message, revStartXMLData message, revStartXMLNode message, revStartXMLTree message, revXMLTrees function

Summary

Creates an XML tree structure from XML text data.

Syntax

revCreateXMLTree(XMLText,parseBadData,createTree,sendMessages)

Example Code

```
revCreateXMLTree(field "XML Data",true,true,false)  
put revCreateXMLTree(theData,false,true,false) into theError
```

Comments

Use the revCreateXMLTree function to make XML text into an XML tree that you can use with other XML library commands and functions.

Parameters:

The XMLText is a string.

The parseBadData is true or false.

The createTree is true or false.

The sendMessages is true or false.

Value:

The revXMLCreateTree function returns a tree ID which can be used to refer to the tree in other XML library commands and functions. The tree ID is always a positive integer. (If the createTree is false, the function returns zero.)

If the function encounters an error while parsing the data, it returns an error message beginning with "xmlerr".

Comments:

If the parseBadData is true, the revXMLCreateTree function tries to parse XML data even if it is not well-formed. Otherwise, the function stops executing as soon as it encounters data that is not well-formed XML.

If the createTree is true, the function creates a tree structure in memory. Otherwise, the function simply parses the XML data without creating an XML tree.

If the sendMessages is true, the revStartXMLTree, revStartXMLNode, revStartXMLData, revEndXMLNode, and revEndXMLTree messages are sent while the XML data is being parsed. Otherwise, these messages are not sent.

If the revXMLCreateTree function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLCreateTree function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revCreateXMLTreeFromFile

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCreateXMLTree function, revDeleteAllXMLTrees command, revDeleteXMLTree command, revEndXMLNode message, revEndXMLTree message, revStartXMLData message, revStartXMLNode message, revStartXMLTree message, revXMLTrees function

Summary

Reads an XML file, optionally creating an XML tree.

Syntax

revCreateXMLTreeFromFile(filePath,parseBadData,createTree,sendMessages)

Example Code

```
revCreateXMLTreeFromFile("New.xml",false,true,false)
if revCreateXMLTreeFromFile(it,true,true,false) then next repeat
```

Comments

Use the revCreateXMLTreeFromFile function to read the contents of an XML document and create an XML tree from it in memory.

Parameters:

The filePath is the name and location of a file containing XML text.

The parseBadData is true or false.

The createTree is true or false.

The sendMessages is true or false.

Value:

The revXMLCreateTreeFromFile function returns a tree ID which can be used to refer to the tree in other XML library commands and functions. The tree ID is always a positive integer. (If the createTree is false, the function returns zero.)

If the function encounters an error while parsing the file, it returns an error message beginning with "xmlerr".

Comments:

If the parseBadData is true, the revCreateXMLTreeFromFile function tries to parse XML data even if it is not well-formed. Otherwise, the function stops executing as soon as it encounters data that is not well-formed XML.

If the createTree is true, the function creates a tree structure in memory. Otherwise, the function simply parses the XML data without creating an XML tree.

If the sendMessages is true, the revStartXMLTree, revStartXMLNode, revStartXMLData, revEndXMLNode, and revEndXMLTree messages are sent while the file is being parsed. Otherwise, these messages are not sent.

If the revCreateXMLTreeFromFile function encounters an error, it returns an error message starting with "xmlerr".

Important! The revCreateXMLTreeFromFile function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revCurrentRecord

function

Synonyms

revdb_currentrecord

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecordIsFirst function, revCurrentRecordIsLast function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, revNumberOfRecords function, About connecting to and using SQL databases, How to create a record set in a SQL database, How to move through the records in a record set (database cursor)

Summary

Returns the number of the current record in a record set (database cursor).

Syntax

revCurrentRecord(recordSetID)

Example Code

```
revCurrentRecord(bookSearchResults)
put revCurrentRecord(it) into myRecordNumber
```

Comments

Use the revCurrentRecord function to find out which record is currently being used.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Value:

The revCurrentRecord function returns a number.

Comments:

If the operation is not successful, the revCurrentRecord function returns an error message that begins with the string "revdberr".

Important! The revCurrentRecord function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revCurrentRecord synonym was added in version 2.0.

revCurrentRecordIsFirst

function

Synonyms

revdb_isbof

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revCurrentRecordIsLast function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, revNumberOfRecords function, About connecting to and using SQL databases, How to create a record set in a SQL database, How to move through the records in a record set (database cursor)

Summary

Returns whether the current record is the first record in a record set (database cursor).

Syntax

revCurrentRecordIsFirst(recordSetID)

Example Code

```
revCurrentRecordIsFirst(line 2 of field ID 2345)  
set the hilite of button "First" to revCurrentRecordIsFirst(it)
```

Comments

Use the revCurrentRecordIsFirst function to stop when you reach the beginning of the records.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Value:

The revCurrentRecordIsFirst function returns true if there are no previous records, false otherwise.

Comments:

If the operation is not successful, the revCurrentRecordIsFirst function returns an error message that begins with the string "revdberr".

Important! The revCurrentRecordIsFirst function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revCurrentRecordIsFirst synonym was added in version 2.0.

revCurrentRecordIsLast

function

Synonyms

revdb_iseof

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revCurrentRecordIsFirst function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, revNumberOfRecords function, About connecting to and using SQL databases, How to create a record set in a SQL database, How to move through the records in a record set (database cursor)

Summary

Returns whether the current record is the last record in a record set (database cursor).

Syntax

revCurrentRecordIsLast(recordSetID)

Example Code

```
revCurrentRecordIsLast(3)
if revCurrentRecordIsLast(testMySQL) then exit repeat
```

Comments

Use the revCurrentRecordIsLast function to stop when you reach the end of the records.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Value:

The revCurrentRecordIsLast function returns true if there are no more records, false otherwise.

Comments:

If the operation is not successful, the revCurrentRecordIsLast function returns an error message that begins with the string "revdberr".

Important! The revCurrentRecordIsLast function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revCurrentRecordIsLast synonym was added in version 2.0.

revDatabaseColumnCount

function

Synonyms

revdb_columncount

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revDatabaseColumnNames function, revDatabaseColumnNumbered function, revDatabaseColumnTypes function, About connecting to and using SQL databases, How to get the contents of a database field

Summary

Returns the number of database fields in a record set (database cursor).

Syntax

revDatabaseColumnCount(recordSetID)

Example Code

```
revDatabaseColumnCount 12  
revDatabaseColumnCount(the savedResultsPointer of this card)
```

Comments

Use the revDatabaseColumnCount function to find out how many database fields you must deal with in a record set (database cursor).

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Value:

The revDatabaseColumnCount function returns a positive integer.

Comments:

If the operation is not successful, the revDatabaseColumnCount function returns an error message that begins with the string "revdberr".

Important! The revDatabaseColumnCount function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revDatabaseColumnCount synonym was added in version 2.0.

revDatabaseColumnIsNull

function

Synonyms

revdb_columnisnull

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revDatabaseColumnCount function, revDatabaseColumnLengths function, revDatabaseColumnNames function, revDatabaseColumnNamed function, revDatabaseColumnNumbered function, revDatabaseColumnTypes function

Summary

Returns true if the specified database field has a null value, false otherwise.

Syntax

revDatabaseColumnIsNull(recordSetID,columnNumber)

Example Code

```
revDatabaseColumnIsNull(currentResults,the number of items in myString)
repeat while revDatabaseColumnIsNull(12,3) -- is field 3 null?
```

Comments

Use the revDatabaseColumnIsNull function to find out whether one of the database fields in the current record is set to a value.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set (database cursor) was created.

The columnNumber is the number of the column.

Value:

The revDatabaseColumnIsNull function returns true or false.

Comments:

A null value in a SQL database refers to a value that is unknown or not applicable. For example, if the price of an item in a product database has not yet been determined, its "Price" field is typically set to null. The null value is not zero: the item's price is not zero, but undetermined.

The special null value allows the database to differentiate between an unknown value deliberately inserted, and a value of zero or a blank field.

This means that if the data value of the database field is empty or zero, the revDatabaseColumnIsNull function returns true. It only returns false if the field's value is not set at all.

Important! The revDatabaseColumnIsNull function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revDatabaseColumnIsNull synonym was added in version 2.0.

revDatabaseColumnLengths

function

Synonyms

revdb_columnlengths

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revDatabaseColumnNamed function, revDatabaseColumnNumbered function, revDatabaseColumnIsNull function, revDatabaseColumnTypes function, About connecting to and using SQL databases

Summary

Returns the maximum field sizes in a record set (database cursor).

Syntax

revDatabaseColumnLengths(recordSetID)

Example Code

```
revDatabaseColumnLengths(storedSet)
```

Comments

Use the revDatabaseColumnLengths function to find out how large a database field is.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set (database cursor) was created.

Value:

The revDatabaseColumnLengths function returns a list of maximum lengths, separated by commas.

Comments:

Each item in the revDatabaseColumnLengths is the maximum size of a database field.

If the operation is not successful, the revDatabaseColumnLengths function returns an error message that begins with the string "revdberr".

Important! The revDatabaseColumnLengths function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revDatabaseColumnLengths synonym was added in version 2.0.

revDatabaseColumnNamed

function

Synonyms

revdb_columnbyname

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revDatabaseColumnNames function, revDatabaseColumnNumbered function, revDatabaseColumnIsNull function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, About connecting to and using SQL databases

Summary

Returns data from a specified database field in a database.

Syntax

revDatabaseColumnNamed(recordSetID,columnName[,holderVariable])

Example Code

```
revDatabaseColumnNamed(myResults,"LASTNAME")
revDatabaseColumnNamed(zipSearch,"CARRIER",foundCarriers)
```

Comments

Use the revDatabaseColumnNamed function to obtain the contents of a given database field, in the current record of the specified record set (database cursor).

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

The columnName is the name of a database field.

The holderVariable is any valid variable name.

Value:

The revDatabaseColumnNamed function returns the text or binary data from the specified field of the current record, if no holderVariable is specified.

Comments:

If you specify a holderVariable, the data is placed in that variable. Otherwise, the data is returned by the function.

If the attempt to get the data is not successful, the revDatabaseColumnNamed function returns an error message that begins with the string "revdberr".

Important! The revDatabaseColumnNamed function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revDatabaseColumnNamed synonym was added in version 2.0.

revDatabaseColumnNames

function

Synonyms

revdb_columnnames

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revDatabaseColumnNamed function, revDatabaseColumnTypes function, About connecting to and using SQL databases

Summary

Returns the list of database field names in a record set (database cursor).

Syntax

revDatabaseColumnNames(recordSetID)

Example Code

```
revDatabaseColumnNames(foundResults)
get revDatabaseColumnNames(field "Current Results")
```

Comments

Use the revDatabaseColumnNames function to find out what database fields are in the record set returned by a SQL query.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Value:

The revDatabaseColumnNames function returns a list of database field names, separated by commas.

Comments:

If the operation is not successful, the revDatabaseColumnNames function returns an error message that begins with the string "revdberr".

Important! The revDatabaseColumnNames function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revDatabaseColumnNames synonym was added in version 2.0.

revDatabaseColumnNumbered function

Synonyms

revdb_columnbynumber

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revDatabaseColumnNamed function, revDatabaseColumnCount function, revDatabaseColumnIsNull function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, About connecting to and using SQL databases

Summary

Returns data from a specified database field.

Syntax

revDatabaseColumnByNumber(recordSetID,columnNumber[,variable])

Example Code

```
revDatabaseColumnByNumber(customerServiceDB,3)
revDatabaseColumnByNumber(7,3,newColumn) -- puts column 3 into newColumn
revDatabaseColumnByNumber(USCustomers,x,foundZipCodes)
```

Comments

Use the revDatabaseColumnNumbered function to obtain the contents of a given database field, in the current record of the specified record set (database cursor).

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

The columnNumber is the number of a database field.

The holderVariable is any valid variable name.

Value:

The revDatabaseColumnNumbered function returns the binary or text data from the specified field of the current record, if no holderVariable is specified.

Comments:

If you specify a holderVariable, the data is placed in that variable. Otherwise, the data is returned by the function.

If the attempt to get the data is not successful, the revDatabaseColumnNumbered function returns an error message that begins with the string "revdberr".

Important! The revDatabaseColumnNumbered function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revDatabaseColumnNumbered synonym was added in version 2.0.

revDatabaseColumnTypes

function

Synonyms

revdb_columntypes

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revDatabaseColumnCount function, revDatabaseColumnIsNotNull function, revDatabaseColumnLengths function, revDatabaseColumnNames function, revDatabaseType function, About connecting to and using SQL databases

Summary

Returns the data types of the columns in a record set (database cursor).

Syntax

revDatabaseColumnDataTypes(recordSetID)

Example Code

```
revDatabaseColumnTypes(the savedCursor of field 1)
if "BLOB" is among the items of revDatabaseColumnTypes(myCursor)
    then put true into preferQueryBlob
```

Comments

Use the revDatabaseColumnTypes function to find out the data type of a database field.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Value:

The revDatabaseColumnTypes function returns a list of data types, one for each column in the record set, separated by commas.

Comments:

The possible data types are:

- BIT
- CHAR
- STRING
- WSTRING

- ï BLOB (short for binary large object)
- ï TIMESTAMP
- ï DATE
- ï TIME
- ï DATETIME
- ï FLOAT
- ï DOUBLE
- ï INTEGER
- ï SMALLINT
- ï WORD
- ï BOOLEAN
- ï LONG

You can use the `revDatabaseColumnTypes` function to find out what kind of data you need to be prepared for. For example, if BLOB is one of the data types, that column is binary data, so you should not attempt to display it in a field.

Important! The `revDatabaseColumnTypes` function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The `revDatabaseColumnTypes` synonym was added in version 2.0.

revDatabaseConnectResult

function

Synonyms

revdb_connectionerr

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCloseDatabase command, revOpenDatabase function, revQueryResult function, About connecting to and using SQL databases

Summary

Returns the most recent error message for a database.

Syntax

revDatabaseConnectResult(databaseID)

Example Code

```
revDatabaseConnectResult(line x of the openDatabases)
if revDatabaseConnectResult(myReservations) is empty then exit repeat
```

Comments

Use the revDatabaseConnectResult function to check for successful completion of commands.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

Value:

The revDatabaseConnectResult function returns a string.

Comments:

If there were no errors on the specified database connection, the revDatabaseConnectResult function returns empty.

Important! The revDatabaseConnectResult function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revDatabaseConnectResult synonym was added in version 2.0.

revDatabaseCursors

function

Synonyms

revdb_cursors

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCloseCursor command, revOpenDatabases function, revDatabaseID function, revQueryResult function, About connecting to and using SQL databases, How to create a record set in a SQL database, How to find out which record sets are open in a database

Summary

Returns the record set (database cursor) IDs associated with a connection to a database.

Syntax

revDatabaseCursors(databaseID)

Example Code

```
revDatabaseCursors(weeklyReportsDB)
put the number of items of revDatabaseCursors(2) into numberOfSets
repeat for each item thisResultSet in revDatabaseCursors(myDB)
```

Comments

Use the revDatabaseCursors function to perform an operation on each record set, or to find out how many record sets exist on a particular database connection.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

Value:

The revDatabaseCursors function returns a list of record set IDs separated by commas.

Comments:

The record set IDs returned are those opened by the specified databaseID.

Important! The revDatabaseCursors function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revDatabaseCursors synonym was added in version 2.0.

revDatabaseID

function

Synonyms

revdb_cursorconnection

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCloseDatabase command, revOpenDatabase function, About connecting to and using SQL databases

Summary

Returns the database ID of the database that opened a record set (database cursor).

Syntax

revDatabaseID(recordSetID)

Example Code

```
revDatabaseID(weeklyOccupancy)
if revDatabaseID(currentCursor) is not field "DB ID" then next repeat
```

Comments

Use the revDatabaseID function to find out which database a record set belongs to.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Value:

The revDatabaseID function returns a positive integer.

Comments:

The returned value is the same as the value returned by the revOpenDatabase function when the database was first opened.

You can use the returned value to close a database after your stack is finished with it.

Important! The revDatabaseID function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revDatabaseID synonym was added in version 2.0.

revDatabaseType

function

Synonyms

revdb_dbtype

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revDatabaseColumnTypes function, revOpenDatabases function, About connecting to and using SQL databases, Database Types Reference

Summary

Returns the type of database associated with a connection.

Syntax

revDatabaseType(databaseID)

Example Code

```
if revDatabaseType(1) is not "odbc" then doCheckLicense
if revDatabaseType(the currentDB of group "Display") is "Valentina"

    then sendValQuery else sendGenericQuery
```

Comments

Use the revDatabaseType function to send different queries or perform different operations depending on what kind of database is being used.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

Value:

The revDatabaseType function returns one of the following strings: "Oracle", "MySQL", "ODBC", "PostgreSQL", or "Valentina".

Comments:

You can use the revDatabaseType function to distinguish between different database types. For example, you may need to frame a SQL query differently depending on what type of database you are communicating with.

Important! The revDatabaseType function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revDatabaseType synonym was added in version 2.0.

revDataFromQuery

function

Synonyms

revdb_querylist

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCloseCursor command, revOpenDatabase function, revExecuteSQL command, revQueryDatabase function, revQueryDatabaseBLOB function, About connecting to and using SQL databases, How to get the contents of a database field, How to get the contents of records in a SQL database

Summary

Gets records from a database according to a SQL query and places the resulting data in a variable, without creating a record set (database cursor).

Syntax

revDataFromQuery([columnDelim],[rowDelim],databaseID,SQLQuery[,varsList])

Example Code

```
revDataFromQuery(,currentDB,field "Query") -- first 2 params are empty  
revDataFromQuery(comma,return,myConnect,grabAllQuery)
```

Comments

Use the revDataFromQuery function when you want to use or display data from a database, but not continue to work with the records that contain it.

Parameters:

The columnDelim is a character, or an expression that evaluates to a character. If no columnDelim is specified, columns are delimited by the tab character.

The rowDelim is a character, or an expression that evaluates to a character. If no rowDelim is specified, rows are delimited by the return character.

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

The SQLQuery is a string in Structured Query Language.

The varsList consists of one or more variable names (or expressions that evaluate to variable names), separated by commas.

Value:

The revDataFromQuery function returns the data in the records selected by the SQLQuery, with the records delimited by the rowDelim and the database fields within each record delimited by the columnDelim.

Comments:

It is convenient to use the revDataFromQuery function, instead of revDatabaseQuery, when you want to obtain the data for use but don't need to retain a reference to the records that the data came from. The revDataFromQuery function executes the SQLQuery, gets the records found by the query, closes the record set created by the query, and returns the data.

The SQLQuery may contain one or more placeholders, which are sequential numbers prepended by a colon. The revDataFromQuery function substitutes the corresponding variable name in the variablesList for each of these placeholders. For example, if you have two variables called "valueX" and "valueY", you can use a SQLQuery that includes placeholders as follows:

```
get revdb_query(myID,"insert into emp() values(:1,:2,:1)",  
"valueX","valueY")
```

The content of the variable valueX is substituted for the ":1" in the SQLQuery (in both places where ":1" appears), and the content of valueY is substituted for ":2".

To pass binary data in a variable in the variablesList, prepend "*b" to the variable name. The revDataFromQuery function strips the binary marker "*b" and passes it to the database as binary data, rather than text data.

You can also use the name of a numerically indexed array, instead of a list of variable names. In this case, the elements of the array are substituted for the corresponding placeholders. To pass binary data in an array element, prepend "*b" to the element's key.

To pass an asterisk as part of the query, substitute a percent sign (%). For example, to use the query "SELECT * FROM Customers WHERE Cust.Name Like '*Inc.'", use a statement like the following:

```
get revDataFromQuery(2,  
"SELECT * FROM Customers WHERE Cust.Name Like '%Inc.'")
```

If the query is not successful, the revDataFromQuery function returns an error message that begins with the string "revdberr.". You can test for success by checking whether the first item of the returned value is "revdberr".

Important! The revDataFromQuery function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revDataFromQuery synonym was added in version 2.0.

revdb_closecursor function

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCloseCursor command, revCloseDatabase command, revDatabaseCursors function, About connecting to and using SQL databases

Summary

Executes the revCloseCursor command.

Syntax

revdb_closecursor(recordSetID)

Example Code

```
revdb_closeCursor(3)
revdb_closeCursor(finalResults)
```

Comments

Use the revdb_closecursor function to clean up after using a record set (database cursor).

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Value:

The revdb_closecursor function returns an error message if the closure is not successful. Otherwise, it returns empty.

Comments:

Evaluating the revdb_closecursor function has the same effect as executing the revCloseCursor command. The only difference is that one is a function call and the other is a command.

Important! The revdb_closecursor function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revdb_commit

function

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCommitDatabase command, revQueryResult function, revRollBackDatabase command, About connecting to and using SQL databases

Summary

Executes the revCommitDatabase command.

Syntax

revdb_commit(databaseID)

Example Code

```
get revdb_commit(currentDB)
```

Comments

Use the revdb_commit function to commit changes.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

Value:

The revdb_commit function returns empty if the changes were successfully saved.

Comments:

Evaluating the revdb_commit function has the same effect as executing the revCommitDatabase command. The only difference is that one is a function call and the other is a command.

Important! The revdb_commit function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revdb_disconnect

function

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCloseCursor command, revCloseDatabase command, revOpenDatabase function, About connecting to and using SQL databases

Summary

Executes the revCloseDatabase command.

Syntax

revdb_disconnect(databaseID)

Example Code

```
get revdb_disconnect(savedBudgetDB)
```

Comments

Use the revdb_disconnect function to cleanly log off from a database.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

Value:

The revdb_disconnect function returns empty if the closure was successful.

Comments:

Evaluating the revdb_disconnect function has the same effect as executing the revCloseDatabase command. The only difference is that one is a function call and the other is a command.

Important! The revdb_disconnect function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revdb_execute function

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revExecuteSQL command, revOpenDatabase function, revQueryDatabase function, revQueryDatabaseBLOB function, About connecting to and using SQL databases

Summary

Executes the revExecuteSQL command.

Syntax

revdb_execute(databaseID,SQLQuery[,variablesList])

Example Code

```
revdb_execute(myDatabaseID,field "Query","*b" & "myvar")
```

Comments

Use the revdb_execute function to execute a SQL query without selecting records.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

The SQLQuery is a string in Structured Query Language.

The variablesList consists of one or more variable names (or expressions that evaluate to variable names), separated by commas.

Value:

The revdb_execute function returns the number of rows operated on by the SQLQuery.

Comments:

Evaluating the revdb_execute function has the same effect as executing the revExecuteSQL command. The only difference is that one is a function call and the other is a command.

Important! The `revdb_execute` function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revdb_movefirst function

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revCurrentRecordIsFirst function, revCurrentRecordIsLast function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, revNumberOfRecords function, About connecting to and using SQL databases

Summary

Executes the revMoveToFirstRecord command.

Syntax

revdb_movefirst(recordSetID)

Example Code

```
revdb_movefirst(savedResults)
```

Comments

Use the revdb_movefirst function to move around within the selected set of records.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set (database cursor) was created.

Value:

The revdb_movefirst function returns true if it successfully moved to the first record, false if it could not move to the first record because there are no records in the record set.

Comments:

Evaluating the revdb_movefirst function has the same effect as executing the revMoveToFirstRecord command. The only difference is that one is a function call and the other is a command.

Important! The revdb_movefirst function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revdb_movelastr function

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revCurrentRecordIsFirst function, revCurrentRecordIsLast function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, revNumberOfRecords function, About connecting to and using SQL databases

Summary

Executes the revMoveToLastRecord command.

Syntax

revdb_movelastr(recordSetID)

Example Code

```
revdb_movelastr(mySearch)
```

Comments

Use the revdb_movelastr function to move around within the selected set of records.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set (database cursor) was created.

Value:

The revdb_movelastr function returns true if it successfully moved to the last record, false if it could not move to the last record because there are no records in the record set.

Comments:

Evaluating the revdb_movelastr function has the same effect as executing the revMoveToLastRecord command. The only difference is that one is a function call and the other is a command.

Important! The revdb_movelastr function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revdb_movenext function

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revCurrentRecordIsFirst function, revCurrentRecordIsLast function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, revNumberOfRecords function, About connecting to and using SQL databases

Summary

Executes the revMoveToNextRecord command.

Syntax

revdb_movenext(recordSetID)

Example Code

```
revdb_movenext(line x of field "Current Recordsets")
```

Comments

Use the revdb_movenext function to move around within the selected set of records.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set (database cursor) was created.

Value:

The revdb_movenext function returns true if it successfully moved to the next record, false if it could not move to the next record because it's already at the end.

Comments:

Evaluating the revdb_movenext function has the same effect as executing the revMoveToNextRecord command. The only difference is that one is a function call and the other is a command.

Important! The revdb_movenext function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revdb_moveprev function

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revCurrentRecordIsFirst function, revCurrentRecordIsLast function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, revNumberOfRecords function, About connecting to and using SQL databases

Summary

Executes the revMoveToPreviousRecord command.

Syntax

revdb_moveprev(recordSetID)

Example Code

```
revdb_moveprev(testresults)
```

Comments

Use the revdb_moveprev function to move around within the selected set of records.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set (database cursor) was created.

Value:

The revdb_moveprev function returns true if it successfully moved to the previous record, false if it could not move to the next record because it's already at the beginning.

Comments:

Evaluating the revdb_moveprev function has the same effect as executing the revMoveToPreviousRecord command. The only difference is that one is a function call and the other is a command.

Important! The revdb_moveprev function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revdb_rollback

function

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCommitDatabase command, revCloseDatabase command, revert command, revRollbackDatabase command, About connecting to and using SQL databases

Summary

Executes the revRollBackDatabase command.

Syntax

revdb_rollback(databaseID)

Example Code

```
revdb_rollback(monthlyExpensesConnection)
```

Comments

Use the revdb_rollback function to discard changes to a database.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

Value:

The revdb_rollback function returns empty if the changes were successfully rolled back.

Comments:

Evaluating the revdb_rollback function has the same effect as executing the revRollBackDatabase command. The only difference is that one is a function call and the other is a command.

Important! The revdb_rollback function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revDeleteAllXMLTrees

command

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCreateXMLTree function, revCreateXMLTreeFromFile function, revDeleteXMLTree command, revXMLTrees function

Summary

Removes all XML tree structures in memory.

Syntax

revDeleteAllXMLTrees

Example Code

```
revDeleteAllXMLTrees  
if revXMLTrees() is not empty then revDeleteAllXMLTrees
```

Comments

Use the revDeleteAllXMLTrees command to free up memory after you finish using the XML library.

Comments:

If the revDeleteAllXMLTrees command encounters an error, the result is set to an error message beginning with "xmlerr".

Important! The revDeleteAllXMLTrees command is part of the XML library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revDeleteFolder

command

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer folder command, create folder command, delete file command, folders function, revCopyFolder command, revMoveFolder command, About filename specifications and file paths, How to investigate the Revolution custom libraries, How to list the contents of a folder

Summary

Removes a folder and all its contents.

Syntax

revDeleteFolder folderToDelete

Example Code

```
revDeleteFolder "/data/Backup Folder"  
revDeleteFolder myTempFolder
```

Comments

Use the revDeleteFolder function to remove a folder from the disk.

Parameters:

The folderToDelete specifies the name and location of the folder. If you specify a name but not a location, Revolution assumes the folder is in the defaultFolder.

Comments:

The revDeleteFolder command removes the entire folder, including all files, subfolders, and their contents.

The revDeleteFolder command uses system services on each platform to perform the move. On Mac OS and OS X systems, it uses AppleScript; on Windows and Unix systems, it uses the shell function. Any errors encountered are returned in the result function.

On Mac OS and OS X systems, the revDeleteFolder command places the folder in the Trash.

Caution! This command can be used to rename or move files and folders your stack did not create. Of course, a stack should not rename or move files and folders it didn't create without obtaining explicit confirmation from the user.

Important! The `revDeleteFolder` command is part of the Common library. To ensure that the command works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The `revDeleteFolder` command is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revDeleteFolder` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revDeleteFolder` command can be used in any handler.

revDeleteXMLNode

command

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revAddXMLNode command, revDeleteXMLTree command, revEndXMLNode message, revPutIntoXMLNode command, revStartXMLNode message, revXMLRootNode function, revXMLNodeContents function

Summary

Removes a node from an XML tree.

Syntax

revDeleteXMLNode treeID,nodeToDelete

Example Code

```
revDeleteXMLNode 3,"/Works/Books/Fiction"  
revDeleteXMLNode line 3 of theTrees,currNodePath
```

Comments

Use the revDeleteXMLNode command to remove part of an XML tree you're working with.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The nodeToDelete is the path to the node you want to remove.

Comments:

If the revDeleteXMLNode command encounters an error, the result is set to an error message beginning with "xmlerr".

Important! The revDeleteXMLNode command is part of the XML library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revDeleteXMLTree

command

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCreateXMLTree function, revCreateXMLTreeFromFile function, revDeleteAllXMLTrees command, revDeleteXMLNode command, revXMLTrees function

Summary

Removes an XML tree structure from memory.

Syntax

revDeleteXMLTree treeID

Example Code

```
revDeleteXMLTree 2  
revDeleteXMLTree the currTree of this card
```

Comments

Use the revDeleteXMLTree command to free up memory when you're finished using an XML tree.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

Comments:

If the revDeleteXMLTree command encounters an error, the result is set to an error message beginning with "xmlerr".

Important! The revDeleteXMLTree command is part of the XML library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revEndXMLNode

message

Synonyms

Objects

card

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revAddXMLNode command, revCreateXMLTree function, revCreateXMLTreeFromFile function, revDeleteXMLNode command, revStartXMLNode message, revXMLEndTree message

Summary

Sent to the current card when the revCreateXMLTreeFromFile function encounters a closing tag while parsing an XML file.

Syntax

revEndXMLNode

Example Code

```
on revEndXMLNode -- notify user how many nodes have been parsed
  global nodesProcessed
  add 1 to nodesProcessed
  put nodesProcessed && "nodes processed" into field "Progress"
end revEndXMLNode
```

Comments

Handle the revEndXMLNode message if you want to build your own subset of an XML document.

Comments:

The revCreateXMLTree or revCreateXMLTreeFromFile functions take XML data and parse it. When you call either of these functions, you can specify whether or not to send messages during the parsing operation.

If you have specified that you want the function to send messages, the revEndXMLNode message is sent when the function encounters a close tag. If you have specified that you don't want messages sent, no revEndXMLNode messages are sent.

revEndXMLTree

message

Synonyms

Objects

card

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCreateXMLTree function, revCreateXMLTreeFromFile function, revEndXMLNode message, revStartXMLTree message

Summary

Sent to the current card when the revCreateXMLTreeFromFile function finishes parsing an XML document.

Syntax

revEndXMLTree

Example Code

```
on revEndXMLTree
    save this stack -- save stack with XML data to disk
end revEndXMLTree
```

Comments

Handle the revEndXMLTree message if you want to build your own subset of an XML document.

Comments:

The revCreateXMLTree or revCreateXMLTreeFromFile functions take XML data and parse it. When you call either of these functions, you can specify whether or not to send messages during the parsing operation.

If you have specified that you want the function to send messages, the revEndXMLTree message is sent when the function encounters the final tag in the file. If you have specified that you don't want messages sent, no revEndXMLTree messages are sent.

reverse

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the inkMode of graphic "Inversion" to reverse
```

Comments

Use the reverse keyword to invert the color under an object.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed. When the reverse mode is used, each component of the color underneath the object (red, green, and blue) is inverted, and the inverse is used for the displayed color.

If the color is expressed as three integers between zero and 255 (one for each of red, green, and blue) then the inverse of each component is equal to 255 minus the component's value. The color of the object itself does not affect the final color.

For example, suppose an object's color is 90,70,250, and the color of a pixel under the object is 60,250,100. If the reverse mode is used, the pixel's displayed color is 195,5,155. (255 minus 60 is 195, 255 minus 250 is 5, and 255 minus 100 is 155.)

revert

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

mainStack property, save command, subStacks property, File menu > Revert to Saved...

Summary

Restores the current stack to its state when last saved, deleting all changes made since the last save.

Syntax

revert

Example Code

```
revert  
if it is "Don't Save" then revert
```

Comments

Use the revert command to back out of any changes made to the current stack since the last save.

Comments:

The revert command also undoes changes made to other stacks stored in the same stack file. That is, if you revert a main stack, all substacks of that stack also revert to the last save, and if you revert a substack, its main stack and any other substacks also revert.

revExecuteSQL

command

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revOpenDatabase function, revQueryDatabase function, revQueryDatabaseBLOB function, About connecting to and using SQL databases, How to create a record set in a SQL database, How to execute a SQL statement on a database

Summary

Executes a SQL statement on a database.

Syntax

revExecuteSQL databaseID,SQLStatement[, {variablesList | arrayName}]

Example Code

```
revExecuteSQL myDatabaseID,field "Query","*b" & "myvar"  
revExecuteSQL 12,builtQuery,"someArray"
```

Comments

Use the revExecuteSQL command to execute a SQL query without selecting records.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

The SQLStatement is a string in Structured Query Language. (Do not include a semicolon at the end of the SQLStatement.)

Note: Some database implementations, such as Oracle, are capable of handling a multiple-line SQL statement.

The variablesList consists of one or more variable names (or expressions that evaluate to variable names), separated by commas.

The arrayName is the name of a single array variable whose keys are sequential numbers.

Note: The variable names or arrayName must be enclosed in quotes; otherwise, the variable's value rather than its name is passed to the revExecuteSQL command.

Comments:

The SQLStatement may contain one or more placeholders, which are sequential numbers prepended by a colon. The revExecuteSQL command substitutes the corresponding item in the variablesList for each of these placeholders. For example, if you have two variables called "valueX" and "valueY", you can use a SQLStatement that includes placeholders as follows:

```
revExecuteSQL myID,"insert into emp() values(:1,:2,:1)",  
"valueX","valueY"
```

The content of the variable valueX is substituted for the ":1" in the SQLQuery (in both places where ":1" appears), and the content of valueY is substituted for ":2".

If you specify an arrayName rather than a list of ordinary variables, the revExecuteSQL command substitutes the corresponding element of the array for each of the placeholders in the query:

```
revExecuteSQL myID,"insert into emp() values(:1,:2,:1)","myArray"
```

The content of the element myArray[1] is substituted for the ":1" in the SQLQuery (in both places where ":1" appears), and the content of myArray[2] is substituted for ":2".

To pass binary data in a variable in the variablesList, prepend "*b" to the variable name. The revExecuteSQL command strips the binary marker "*b" and passes it to the database as binary data, rather than text data.

To pass binary data in an array element, prepend "*b" to the element's key.

The SQLStatement should not contain a SELECT statement. (To select records, use the revQueryDatabase function instead.) If it does, or if the operation is not successful, an error message is returned by the result function.

Important! The revExecuteSQL command is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revGoToFramePaused

command

Synonyms

Objects

Animation library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

paused property, revPlayAnimation command, revStopAnimation command, stop command, Animation Builder Tutorial, How to investigate the Revolution custom libraries, Why don't animations run in my standalone?, Tools menu > Animation Builder

Summary

Moves an animation created in the Animation Builder to the specified frame.

Syntax

revGoToFramePaused animationName,{keyFrameName | frameNumber}

Example Code

```
revGoToFramePaused "Fireworks",25 -- go to frame 25
revGoToFramePaused (field "Current Animation"),(field "current Frame")
```

Comments

Use the revGoToFramePaused command to move an animation to its starting frame before playing, or to restore it to a frame after playing.

Parameters:

The animationName is an expression that evaluates to the name of an animation on the current card.

The keyFrameName is the name of a key frame in the animation.

The frameNumber is the number of any frame.

Comments:

You create and name animations using the Animation Builder. To open the Animation Builder, select the object or objects you want to animate, then choose Tools menu Animation Builder. When you go to a frame, the objects in the animation move to their position for that frame.

The animationName must be an animation that was created on the current card of the defaultStack.

To play an animation while allowing other handlers to execute, use the `revGoToFramePause` command repeatedly for each frame, using the `send` command:

```
on goNextFrame theFrame -- 1 frame every 20th of a second
  revGoToFramePaused "My Animation",theFrame
  send ("goNextFrame" && (theFrame + 1)) to me in 3 ticks
end goNextFrame
```

Important! The `revGoToFramePaused` command is part of the Animation library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Animation Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Animation library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revGoToFramePaused` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revGoToFramePaused` command can be used in any handler.

revGoURL

command

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

load command, revMail command, URL keyword, How to investigate the Revolution custom libraries, How to open a URL in a browser

Summary

Opens a URL in a web browser.

Syntax

revGoURL URL

Example Code

```
revGoURL "http://www.example.org/info.html"
revGoURL "http://me:secret@www.example.com/secret-plan/"
revGoURL it
revGoURL "mailto:guido@sales.example.com"
revGoURL "file:///Folder/file.html"
```

Comments

Use the revGoURL command to open a web page in the user's browser application from within a stack.

Parameters:

The URL is an expression that evaluates to a URL.

Comments:

On Mac OS and OS X systems, the URL is opened in the application the user has set for that URL protocol in the Internet control panel (or, for older system versions, Internet Config). For example, "mailto:" URLs are opened in the email program, "http:" URLs are opened in the web browser, and so on.

Important! On Mac OS systems, the revGoURL command requires that AppleScript be installed and that the GURL Apple Event be supported. GURL support is included in Mac OS 8.5 or later, and is available for Mac OS 8.0 and 8.1 if the "Internet Scripting" file is installed in the "Scripting Additions" folder inside the Extensions folder. (This file is installed by the Internet Access installer on the 8.0 installation CD.) The revGoURL command is not supported for Mac OS versions before 8.0.

On Windows systems, the URL is opened in the browser set in the Windows registry.

On Unix systems, the URL is opened in the browser specified in the global variable "gREVWebBrowser". You can set this variable to either a shell command used to launch the browser, or the full path to the browser application. By default, the Mozilla browser is used.

You can specify a "file:" URL. The file will be opened in the user's web browser; usually, such files are HTML files. (It is not necessary to escape or URLEncode file paths that contain spaces.)

Important! The revGoURL command is part of the Common library. To ensure that the command works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The revGoURL command is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the revGoURL command is not yet available. This may affect attempts to use this command in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the revGoURL command can be used in any handler.

Changes to Transcript:

Support for using a "file:" URL and for specifying a browser on Unix systems via the "gREVWebBrowser" global variable was added in version 2.0. In previous versions, Netscape was always used on Unix systems.

revInitializeVideoGrabber

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revCloseVideoGrabber command, revPreviewVideo command, revRecordVideo command, revSetVideoGrabberRect command, revSetVideoGrabSettings command, revVideoGrabSettings function, revVideoGrabDialog command, How to capture video from a video camera, How to change the size and position of the video grabber window, How to change video grabber settings

Summary

Opens the video grabber window.

Syntax

revInitializeVideoGrabber videoMethod,grabberRect

Example Code

```
revInitializeVideoGrabber "QT", "100,100,200,200"  
revInitializeVideoGrabber "VFW", the rect of this stack
```

Comments

Use the revInitializeVideoGrabber command to start up video capture capability.

Parameters:

The videoMethod is either "QT" or "VFW".

The grabberRect is the rectangle of the video grabber window, and consists of four integers separated by commas: the top, left, bottom, and right edges of the video grabber, in absolute (screen) coordinates.

Comments:

You must use the revInitializeVideoGrabber command before using any of the other commands and functions in the Video library. The command does two things:

- ï Loads the code necessary for video capture into memory.
- ï Opens the video grabber window.

Note: The video grabber is not a stack window, so you can't set its properties. To change the size and location of the video grabber, use the `revSetVideoGrabberRect` command.

Once the video grabber is open, you can use the `revVideoGrabDialog` command to specify where the video camera (or other video source) is connected. Use the `revRecordVideo` command to record video from the camera to a file, or use the `revPreviewVideo` command to display video without saving it.

To use QuickTime for video capture (on Mac OS, OS X, or Windows systems), specify "QT" as the `videoMethod`. To use Video for Windows (on Windows systems), specify "VFW".

To close the window and unload the video capture code, you should use the `revCloseVideoGrabber` command when you're done with video capture.

Important! The `revInitializeVideoGrabber` command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revIsSpeaking

function

Synonyms

Objects

Speech library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revSpeak command, sound function

Summary

Returns true or false, depending on whether a phrase is currently being spoken by the revIsSpeaking command.

Syntax

revIsSpeaking()

Example Code

```
revIsSpeaking()  
wait until revIsSpeaking() is false
```

Comments

Use the revIsSpeaking function to avoid interrupting a speech.

Value:

The revIsSpeaking function returns true or false.

Comments:

If you execute the revSpeak command while the computer is already speaking, the first speech is stopped and the second speech begins immediately.

Important! The revIsSpeaking function is part of the Speech library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revLicenseType

function

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

environment function, licensed function, revOpenDatabase function, scriptLimits function, How to investigate the Revolution custom libraries

Summary

Returns the type of license under which the currently running copy of the Revolution development environment is running.

Syntax

revLicenseType()

Example Code

```
if revLicenseType() is "standalone" then explainLimits
if "Enterprise" is not in revLicenseType() then useODBC
```

Comments

Use the revLicenseType function to determine what version of the Revolution development environment is running. Different license types have different capabilities that may affect a stack's operation.

Value:

The revLicenseType function returns the same string as seen in the lower left corner of the splash screen when Revolution starts up.

Comments:

If the development environment is not running, the revLicenseType function returns "standalone".

Important! The revLicenseType function is part of the Common library. To ensure that the function works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The revLicenseType function is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first

part of the application's startup process, before this message is sent, the `revLicenseType` function is not yet available. This may affect attempts to use this function in `startup`, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revLicenseType` function can be used in any handler.

revLoadedStacks

function

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

mainStacks function, openStacks function, reloadStack message, stacks function, How to find out whether a window is open, How to investigate the Revolution custom libraries, Why does Revolution ask to purge a stack?, Recipe for finding out whether a window is open by title

Summary

Returns a list of the names of all stacks that are loaded into memory.

Syntax

revLoadedStacks([whichStacks])

Example Code

```
revLoadedStacks(all) -- includes stacks such as message box
revLoadedStacks(application) -- only includes non-Rev stacks
revLoadedStacks() -- respects current preference setting
```

Comments

Use the revLoadedStacks function if you need to perform some operation on all stacks that are loaded into memory.

Parameters:

The whichStacks is one of the following:

application: Only your application's stacks

all: All stacks including Revolution development environment stacks

preference: Depends on Preferences setting

Value:

The revLoadedStacks function returns a list of the short names of all loaded stacks, one per line.

Comments:

If the whichStacks parameter is all, the revLoadedStacks function returns all stacks that are loaded into memory, including stacks that are part of the Revolution development environment (such as the message box, property inspector, and so on).

If the whichStacks parameter is application, the revLoadedStacks function returns all loaded stacks that are not part of the development environment.

If the whichStacks parameter is preference, the revLoadedStacks function checks the setting of the "Revolution UI Elements in Lists" item in the View menu, and returns either all stacks, or only stacks that are not part of the development environment, depending on the setting.

If you don't specify an includeRevStacks parameter, the revLoadedStacks function returns all stacks that are loaded into memory.

Important! The revLoadedStacks function is part of the Common library. To ensure that the function works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The revLoadedStacks function is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the revLoadedStacks function is not yet available. This may affect attempts to use this function in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the revLoadedStacks function can be used in any handler.

revMacFromUnixPath

function

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

create folder command, delete file command, read from file command, rename command, revUnixFromMacPath function, write to file command, About filename specifications and file paths, How to investigate the Revolution custom libraries

Summary

Converts a Unix-style pathname to a Mac OS-style pathname.

Syntax

revMacFromUnixPath(unixPathname[,convertOSX])

Example Code

```
revMacFromUnixPath("/usr/bin/stuff") -- returns "usr:bin:stuff"  
revMacFromUnixPath(it)  
revMacFromUnixPath(
```

Comments

Use the revMacFromUnixPath function to convert a Revolution-style file path to the Mac OS file path format (for example, if you need to pass a pathname to an external).

Parameters:

The unixPathname is a file or folder pathname in the standard format used by Revolution for file paths.

The convertOSX is true or false. If you don't specify the convertOSX, if OS X is running, Revolution assumes you want to convert an OS X-style path to a Mac OS-style path; otherwise, it assumes you don't want to convert between the OS X style and Mac OS style.

Value:

The revMacFromUnixPath function returns a string with the file path in the format expected by the Mac OS.

Comments:

The `revMacFromUnixPath` function converts slashes (/) to colons (:), the folder-level delimiter for Mac OS pathnames. It also deletes leading slashes, so that pathnames are rooted in the volume name (the standard for Mac OS pathnames). It also adjusts relative pathnames.

On Mac OS systems, absolute paths always begin with the name of the disk that the file or folder is on. On OS X systems, the startup disk's name does not appear in absolute file paths. Instead, if a file or folder is on the startup disk, the first part of the file path is the top-level folder that the file is in. If a file or folder is on a disk other than the startup disk, its absolute path starts with "Volumes", followed by the disk name.

The OS X path convention is used by Revolution, but the old Mac OS-style path convention is required by certain applications (such as AppleScript), even on OS X systems. If the `convertOSX` is true (or if you don't specify the `convertOSX` and the application is running under OS X), the `revMacFromUnixPath` function automatically converts absolute paths from the OS X standard to the Mac OS standard, adding the startup disk's name to paths that are on the startup disk, and stripping the "Volumes" element from paths that are not on the startup disk. If the `convertOSX` is false, the `revMacFromUnixPath` function does not make these changes to absolute paths.

Revolution always uses the Unix pathname standard for cross-platform compatibility, and automatically converts pathnames to the correct standard for the current platform when executing commands. You need to convert the pathname only if you are passing it to another program or external. If you are using only Revolution commands and functions, you do not need to convert the pathname, since Revolution does it for you.

Important! The `revMacFromUnixPath` function is part of the Common library. To ensure that the function works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The `revMacFromUnixPath` function is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revMacFromUnixPath` function is not yet available. This may affect attempts to use this function in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revMacFromUnixPath` function can be used in any handler.

Changes to Transcript:

The `convertOSX` parameter was introduced in version 2.1.1. In previous versions, the `revMacFromUnixPath` function did not attempt to convert between the Mac OS and OS X conventions described above.

revMail

command

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

revGoURL command, How to investigate the Revolution custom libraries

Summary

Opens a new email message in the user's email program.

Syntax

```
revMail address[,ccAddress[,mailSubject[,messageBody]]]
```

Example Code

```
revMail "guido@example.net"  
revMail "help@example.com",,"Help!",field "Message"  
revMail "support@runrev.com","me@example.org","Installation Help"  
revMail the bugAddress of this stack,,"Auto Bug Report",myBugsList
```

Comments

Use the revMail command to create an email message from within a stack.

Parameters:

The address is a string consisting of one or more email addresses, separated by commas. These addresses will appear in the To: header of the email message.

The ccAddress is a string consisting of one or more email addresses, separated by commas. These addresses will appear in the Cc: header of the email message.

The mailSubject is a string consisting of a single line. This text will appear in the Subject: header of the email message.

The messageBody is a string. This text will appear in the body of the email message.

Comments:

When Revolution executes the revMail command, the user's email program is opened (if necessary) and a new email message with the specified parameters is created. The user can change any of the settings

before sending the message, and the message is not sent automatically: the user must explicitly send it (for example, by clicking a "Mail" button in the email program).

On Mac OS systems, the mail message is opened in the application the user has set for the "mailto:" protocol in the Internet control panel (or, for older system versions, Internet Config).

Important! On Mac OS systems, the revMail command requires that AppleScript be installed and that the GURL Apple Event be supported. GURL support is included in Mac OS 8.5 or later, and is available for Mac OS 8.0 and 8.1 if the "Internet Scripting" file is installed in the "Scripting Additions" folder inside the Extensions folder. (This file is installed by the Internet Access installer on the 8.0 installation CD.) The revMail command is not supported for Mac OS versions before 8.0.

For some older email programs, it may not be possible to specify a ccAddress, mailSubject, or messageBody. Generally, the revMail command will still work with such programs, but only the To: header will be set.

Important! The revMail command is part of the Common library. To ensure that the command works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The revMail command is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the revMail command is not yet available. This may affect attempts to use this command in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the revMail command can be used in any handler.

revMoveFolder

command

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer folder command, create alias command, create folder command, delete folder command, folders function, revCopyFolder command, revDeleteFolder command, rename command, specialFolderPath function, About filename specifications and file paths, How to investigate the Revolution custom libraries

Summary

Moves a folder with all its contents to another location.

Syntax

revMoveFolder folderToMove,destinationFolder

Example Code

```
revMoveFolder "Backup Folder", "/Disk/Backups/"  
revMoveFolder "My App Prefs", specialFolderPath(Preferences)
```

Comments

Use the revMoveFolder function to move a folder into another folder.

Parameters:

The folderToMove specifies the name and location of the folder. If you specify a name but not a location, Revolution assumes the folder is in the defaultFolder.

The destinationFolder specifies the name and location of the folder where the folder should be placed. If you specify a name but not a location, Revolution assumes the destination folder is in the defaultFolder.

Comments:

The revMoveFolder command moves the entire folder, including all files, subfolders, and their contents. The folder is removed from its original location and appears only in the new location.

The revMoveFolder command uses system services on each platform to perform the move. On Mac OS and OS X systems, it uses AppleScript; on Windows and Unix systems, it uses the shell function. Any errors encountered are returned in the result function.

Important! The `revMoveFolder` command is part of the Common library. To ensure that the command works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The `revMoveFolder` command is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revMoveFolder` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revMoveFolder` command can be used in any handler.

revMoveToFirstRecord

command

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revCurrentRecordIsFirst function, revCurrentRecordIsLast function, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, revNumberOfRecords function, About connecting to and using SQL databases, How to create a record set in a SQL database, How to move through the records in a record set (database cursor)

Summary

Moves to the first record of a record set (database cursor).

Syntax

revMoveToFirstRecord recordSetID

Example Code

```
revMoveToFirstRecord savedResults
```

Comments

Use the revMoveToFirstRecord command to move around within the selected set of records.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Comments:

If the command could not move to the first record because there are no records in the record set, the result function returns an error message that begins with the string "revdberr".

Note: Not all databases support the revMoveToFirstRecord command.

Important! The revMoveToFirstRecord command is part of the Database library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revMoveToLastRecord

command

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revCurrentRecordIsFirst function, revCurrentRecordIsLast function, revMoveToFirstRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, revNumberOfRecords function, About connecting to and using SQL databases, How to create a record set in a SQL database, How to move through the records in a record set (database cursor)

Summary

Moves to the last record of a record set (database cursor).

Syntax

revMoveToLastRecord recordSetID

Example Code

```
revMoveToLastRecord mySearch
```

Comments

Use the revMoveToLastRecord command to move around within the selected set of records.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Comments:

If the command could not move to the last record because there are no records in the record set, the result function returns an error message that begins with the string "revdberr".

Note: Not all databases support the revMoveToLastRecord command.

Important! The revMoveToLastRecord command is part of the Database library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revMoveToNextRecord

command

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revCurrentRecordIsFirst function, revCurrentRecordIsLast function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToPreviousRecord command, revNumberOfRecords function, About connecting to and using SQL databases, How to create a record set in a SQL database, How to move through the records in a record set (database cursor)

Summary

Moves to the next record in a record set (database cursor).

Syntax

revMoveToNextRecord recordSetID

Example Code

```
revMoveToNextRecord myCurrentItems  
revMoveToNextRecord (line x of field "Current Recordsets")
```

Comments

Use the revMoveToNextRecord command to move around within the selected set of records.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Comments:

If the command could not move to the next record because it's already at the end or there are no records in the record set, the result function returns an error message that begins with the string "revdberr".

Note: Not all databases support the revMoveToNextRecord command.

Important! The revMoveToNextRecord command is part of the Database library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revMoveToPreviousRecord

command

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revCurrentRecordIsFirst function, revCurrentRecordIsLast function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revNumberOfRecords function, About connecting to and using SQL databases, How to create a record set in a SQL database, How to move through the records in a record set (database cursor)

Summary

Moves to the previous record in a record set (database cursor).

Syntax

```
revMoveToPreviousRecord recordSetID
```

Example Code

```
revMoveToPreviousRecord 31  
revMoveToPreviousRecord testResults
```

Comments

Use the revMoveToPreviousRecord command to move around within the selected set of records.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Comments:

If the command could not move to the previous record because it's already at the first record or there are no records in the record set, the result function returns an error message that begins with the string "revdberr".

Note: Not all databases support the revMoveToPreviousRecord command.

Important! The revMoveToPreviousRecord command is part of the Database library. To ensure that the command works in a standalone application, you must include this custom library when you create

your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revNumberOfRecords

function

Synonyms

revdb_recordcount

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCurrentRecord function, revCurrentRecordIsFirst function, revCurrentRecordIsLast function, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, About connecting to and using SQL databases, How to create a record set in a SQL database, How to move through the records in a record set (database cursor)

Summary

Returns the number of records in a record set (database cursor).

Syntax

revNumberOfRecords(recordSetID)

Example Code

```
revNumberOfRecords(westCoastCustomers)
get revNumberOfRecords(15) -- number of records in record set 15
repeat for revNumberOfRecords(pastDueCustomers) times
```

Comments

Use the revNumberOfRecords function to find out how many records matched a query.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Value:

The revNumberOfRecords function returns a non-negative integer.

Comments:

If the query is not successful, the revNumberOfRecords function returns an error message that begins with the string "revdberr".

Important! The revNumberOfRecords function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revNumberOfRecords synonym was added in version 2.0.

revOpenDatabase

function

Synonyms

revdb_connect

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCloseDatabase command, revLicenseType function, revOpenDatabases function, revSetDatabaseDriverPath command, About connecting to and using SQL databases, Database Types Reference, How to close the connection to a database

Summary

Connects to a MySQL, Oracle, ODBC, PostgreSQL, or Valentina database.

Syntax

```
revOpenDatabase(databaseType,host[:port],databaseName,  
  
[userName],[password] [, useSSL |  
  
valentinaCacheSize,valentinaMacSerial,valentinaWindowsSerial])
```

Example Code

```
get revOpenDatabase("MySQL","www.example.com","RatesDB",myUsr,myPass)  
get revOpenDatabase("ODBC","BizFile",,"jenkins",the dbPassword of me)  
get revOpenDatabase("Valentina",,"","::Project:MyVal.vdb",,,, ,  
  
the serialNumber of this card,"")  
get revOpenDatabase("Valentina",,"","G:.vdb",,,, ,  
  
field "Valentina serial")  
get revOpenDatabase("MySQL","www.example.com","RatesDB",myUsr,myPass,0)
```

Comments

Use the revOpenDatabase function to start working with a database.

Parameters:

The databaseType is one of "MySQL","Oracle" (Revolution Enterprise only), "ODBC", "PostgreSQL", or "Valentina".

The host is the IP address or domain name of the system hosting the database. For Valentina databases, which reside on the local system, the host should be empty. If no host is specified, the local system is used.

The port is the port number you want to connect to, and is used only for PostgreSQL and MySQL databases. If no port is specified, MySQL database connections use port 3306 and PostgreSQL database connections use port 5432.

The databaseName is the name of the database to connect to. For Valentina databases, this is the path to the file that contains the database.

Important! When connecting to a Valentina database on a Mac OS, the databaseName must be a Mac-style file path. (To convert from a Revolution file path to the Mac style, use the revMacFromUnixPath function.) When connecting to a Valentina database on a Windows system, the databaseName must be a Windows-style file path. On Unix and OS X systems, you use a standard Revolution file path. (For more information about file paths, see the topic "About filename specifications and file paths".)

The userName is your authorized user name for the database. (Some databases do not require a user name.)

The password is the authentication password for userName. (Some databases do not require a password.)

The useSSL determines whether a connection to a MySQL database uses SSL for the connection. If useSSL is not specified, the connection uses SSL. If the databaseType is not "MySQL", this parameter has no effect.

The valentinaCacheSize is the size in bytes of the database cache used for Valentina databases. The minimum cache size is 524288 (512K). If the valentinaCacheSize is not specified, the database cache is 3 megabytes. If the databaseType is not "Valentina", the valentinaCacheSize has no effect.

The valentinaMacSerial or valentinaWindowsSerial is the serial number that unlocks the Valentina VXCMD. (On Mac OS and OS X systems, specify a valentinaMacSerial and leave the valentinaWindowsSerial parameter empty. On Windows systems, specify a valentinaWindowsSerial and leave the valentinaMacSerial empty.) If no valentinaMacSerial or valentinaWindowsSerial is specified, the database closes automatically after ten minutes.

Value:

The revOpenDatabase function returns a database ID which can be used to refer to the database in other Database library commands and functions. The database ID is always an integer.

Comments:

To use a DSN to identify an ODBC database, use the DSN as the host, and leave the databaseName parameter empty.

When using a Valentina database, specify the valentinaCacheSize for greatest efficiency based on the size of your database. The Valentina cache is an area of memory that the Valentina engine sets aside for database operations. The minimum valentinaCacheSize is 512K, or 524288 bytes. For best efficiency, start with the default cache size of 3 megabytes (3145728 bytes), and add about a megabyte for each 100,000-150,000 records in your database.

When connecting to a Valentina database, the `valentinaMacSerial` and `valentinaWindowsSerial` parameters take effect for the entire session. If you specify a different serial number for subsequent connections, it is ignored until the next time you quit and restart the application.

If the database is not successfully opened, the `revOpenDatabase` function returns an error message. The error message is never an integer, so you can check whether the connection was successful by checking whether the return value is an integer or not.

Important! This function's capabilities depend on your edition of Revolution:

- Revolution Enterprise: (and applications created with Revolution Enterprise): Full access to all database types.

- Other editions: Full access to MySQL, PostgreSQL, and Valentina databases, and via ODBC. Other editions cannot be used for direct access to Oracle databases.

Important! The `revOpenDatabase` function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The ability to specify whether SSL is used for MySQL database connections was added in version 2.1. In version 2.0 and later, SSL was used automatically. In versions before 2.0, SSL was not used.

The `revOpenDatabase` synonym was added in version 2.0.

The ability to connect to PostgreSQL databases was added in version 2.0.

The ability to connect to Valentina databases was added in version 1.1.1.

revOpenDatabases

function

Synonyms

revdb_connections

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revDatabaseConnectResult function, revDatabaseCursors function, revOpenDatabase function, About connecting to and using SQL databases, How to find out which record sets are open in a database

Summary

Returns a list of open databases.

Syntax

revOpenDatabases()

Example Code

```
revOpenDatabases()  
repeat for each item thisConnection in revOpenDatabases()  
if revOpenDatabases is empty then cleanUp
```

Comments

Use the revOpenDatabases function to find out how many databases are open, or in a repeat loop to access all the open databases in turn.

Value:

The revOpenDatabases function returns a list of database IDs separated by commas.

Comments:

The database ID returned by the revOpenDatabases function is the same as the number returned by the revOpenDatabase function when the database was opened.

If the operation is not successful, the revOpenDatabases function returns an error message that begins with the string "revdberr".

Important! The revOpenDatabases command is part of the Database library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revOpenDatabases synonym was added in version 2.0.

revPlayAnimation

command

Synonyms

Objects

Animation library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

play command, revGoToFramePaused command, revStopAnimation command, Animation Builder Tutorial, How to animate a sprite, How to investigate the Revolution custom libraries, Why don't animations run in my standalone?, Tools menu > Animation Builder

Summary

Plays an animation.

Syntax

revPlayAnimation animationName[,startFrame[,endFrame]]

Example Code

```
revPlayAnimation "FlyAroundTheScreen" -- play entire animation
revPlayAnimation myFirstAnim,20 -- start at frame 20
revPlayAnimation (the label of me),1,"End of Intro"
```

Comments

Use the revPlayAnimation command to display an animation that you created in the Animation Builder.

Parameters:

The animationName is an expression that evaluates to the name of an animation on the current card.

The startFrame is either a frame number or the name of a key frame. If no startFrame is specified, the animation starts with the first frame.

The endFrame is either a frame number or the name of a key frame. If no endFrame is specified, the animation continues to the last frame.

Comments:

You create and name animations using the Animation Builder. To open the Animation Builder, select the object or objects you want to animate, then choose Tools menu Animation Builder. When you play the animation, the object or objects in the animation move across the window.

The animationName must be an animation that was created on the current card of the defaultStack.

To play a portion of the animation rather than the whole thing, specify a startFrame and endFrame specifying the part of the animation you want to play.

Important! The revPlayAnimation command is part of the Animation library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Animation Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Animation library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the revPlayAnimation command is not yet available. This may affect attempts to use this command in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the revPlayAnimation command can be used in any handler.

revPreviewVideo

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revInitializeVideoGrabber command, revRecordVideo command, revStopPreviewingVideo command, revVideoGrabIdle command, How to change video grabber settings

Summary

Displays the video input in the video grabber without recording it.

Syntax

revPreviewVideo

Example Code

```
revPreviewVideo  
if the hilite of button "Preview Only" then revPreviewVideo
```

Comments

Use the revPreviewVideo command to see the image from a video camera on the screen.

Comments:

You must use the revInitializeVideoGrabber command to open the video grabber before you can use the revPreviewVideo command.

The revPreviewVideo command shows the input from the video source in the video grabber window, but does not record it. To save the video as a file while displaying it, use the revRecordVideo command instead.

If the video grabber was already recording video to a file, the revPreviewVideo command stops the recording.

Important! If you are using QuickTime for video capture, execute the revVideoGrabIdle command periodically while previewing or recording video. Otherwise, you may experience poor quality in the video capture.

To stop displaying the video input, use the revStopPreviewingVideo command.

Important! The `revPreviewVideo` command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revPrintField

command

Synonyms

Objects

Printing library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer printer command, open printing command, print command, revPrintReport command, revPrintText command, revShowPrintDialog command, How to investigate the Revolution custom libraries, How to print in landscape mode, How to print the contents of a field, Recipe for printing the fields on a card, File menu > Print Field...

Summary

Prints the contents of a field.

Syntax

revPrintField fieldDescriptor

Example Code

```
revPrintField the name of field "Comments"  
revPrintField the long ID of the mouseControl  
revPrintField ("field" && 5) -- prints field 5
```

Comments

Use the revPrintField command to print the formatted contents of a field from within a handler.

Parameters:

The fieldDescriptor is any expression that evaluates to a field reference.

Important! The revPrintField command does not accept direct field references. For example, the following statement causes an error message:

```
revPrintField field "My Field" -- CAN'T USE THIS FORM
```

Instead, use a form that evaluates to a field reference, like this:

```
revPrintField the name of field "My Field" -- use this form instead  
revPrintField ("field" && quote & "My Field" & quote) -- or this
```

Comments:

The revPrintField command is equivalent to selecting the field and choosing File menu Print Field.

If the field contains any expressions of the form <%expression%>, the expression is evaluated and replaced with the value before the field is printed. For example, if the field contains the text

Today's date is <%the long date%>

the printed text reads

Today's date is Friday, February 15, 2002

(This assumes, of course, that the revPrintField command is executed on that date). You can also use the special value <%pageNumber%> in the field: this value is replaced with the page number.

To show the standard print dialog box, use the revShowPrintDialog command before the revPrintField command.

Important! The revPrintField command is part of the Printing library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Printing Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Printing library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the revPrintField command is not yet available. This may affect attempts to use this command in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the revPrintField command can be used in any handler.

revPrintReport

command

Synonyms

Objects

Printing library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

print command, revPrintField command, revPrintText command, How to create a report layout card, How to investigate the Revolution custom libraries, How to specify what cards to include in a printed report, Recipe for printing the fields on a card, Tools menu > Report Builder, Object menu > New Control > Report Object, View menu > Go to Report Page

Summary

Prints a report from the report viewers on the current card.

Syntax

revPrintReport

Example Code

```
revPrintReport
```

Comments

Use the revPrintReport command to print a report that you have set up in the Report Builder.

Comments:

When you use the Report Builder to create a set of report settings, the settings are stored with the card. The revPrintReport command uses the settings specified in the Report Builder for the current card to print the report. If you have never used the Report Builder with the current card (and therefore there are no report settings to use), the revPrintReport command causes a script error.

The revPrintReport command is equivalent to choosing Tools menu Report Builder and clicking "Print Report".

Important! The revPrintReport command is part of the Printing library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Printing Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Printing library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revPrintReport` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revPrintReport` command can be used in any handler.

revPrintText

command

Synonyms

Objects

Printing library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

answer printer command, print command, revPrintField command, revShowPrintDialog command, How to investigate the Revolution custom libraries, How to print in landscape mode, How to print the contents of a field

Summary

Prints formatted or unformatted text.

Syntax

```
revPrintText textToPrint[,headerText[,footerText[,fieldTemplate]]]
```

Example Code

```
revPrintText "Hello world"
revPrintText myCollectedData,"Confidential" -- header with no footer
revPrintText (the htmlText of field "Info"),"Info",the time && the date
revPrintText field 1,,"prepared by John Smith" -- footer, no header
revPrintText it,tab & myTitle -- centered header
revPrintText thisText,,,the long name of field "Text"
```

Comments

Use the revPrintText command to print any text from within a handler.

Parameters:

The textToPrint is an expression that evaluates to a string.

The headerText is an expression that evaluates to a string. If the headerText is empty, no page header is printed.

The footerText is an expression that evaluates to a string. If the footerText is empty, no page footer is printed.

The fieldTemplate is any expression that evaluates to a field reference.

Important! The `revPrintText` command does not accept direct field references for the `fieldTemplate` parameter. For example, the following statement causes an error message:

```
revPrintText myText,,,field "Text" -- CAN'T USE THIS FORM
```

Instead, use a form that evaluates to a field reference, like this:

```
revPrintText myText,,,the name of field "Text" -- use this form
revPrintText myText,,,"(field" && quote & "Text" & quote) -- or this
```

Comments:

The `revPrintText` command can be used to print either formatted text (via the `htmlText` property's format) or plain text. If you are generating formatted text, see the `htmlText` property for a description of the differences between the `htmlText` property's formatting and standard HTML.

If the `textToPrint`, `headerText`, or `footerText` contains `<p>` or a start/end tag pair, the `revPrintText` command assumes the text is in the same format as the `htmlText` property. Otherwise, the `revPrintText` command assumes the text is plain text.

If the text being printed is plain text, and a `fieldTemplate` is specified, the text is printed with that field's `textFont`, `textSize`, and `textStyle`. Otherwise, the text is printed in plain 12-point, with the default font of the platform being used. (The field specified in the `fieldTemplate` does not need to contain any text; only its appearance properties are used, not its contents. If you want to print the contents of the field, use the `revPrintField` command instead.)

If the `textToPrint`, `headerText`, or `footerText` contains any expressions of the form `<%expression%>`, the expression is evaluated and replaced with the value before the text is printed. For example, if the `textToPrint` contains the text

```
Today's date is <%the long date%>
```

the printed text reads

```
Today's date is Friday, February 15, 2002
```

(assuming, of course, that the `revPrintText` command is executed on that date).

You can also use the special values `<%pageNumber%>` and `<%numPages%>` in the `textToPrint`, `headerText`, or `footerText`. These expressions are replaced with the current page number or total number of pages respectively.

The `headerText` and `footerText` can contain up to two tab characters:

- ī Everything up to the first tab character is left-aligned.
- ī Everything between the first and second tab characters is centered.
- ī Everything after the second tab character is right-aligned.

If the `headerText` or `footerText` contain more than two tab characters, everything after the third tab is ignored.

To show the standard print dialog box, use the `revShowPrintDialog` command before the `revPrintText` command.

Tip: If the `textToPrint` contains tags, the tags are interpreted as text style information. To print tags literally, set the text of a field to the text you want to print, then use the `htmlText` of that field as the `textToPrint`. Converting to the `htmlText` escapes the tags and allows them to be printed.

Important! The `revPrintText` command is part of the Printing library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Printing Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Printing library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revPrintText` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revPrintText` command can be used in any handler.

Changes to Transcript:

The `fieldTemplate` parameter was added in version 2.0.

revProfile

property

Synonyms

cRevGeneral["profile"]

Objects

Profile library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

customPropertySets property, revSetCardProfile command, revSetStackFileProfile command, revSetStackProfile command, About properties and property profiles, Property Profiles Tutorial, How to change an object's property profile

Summary

Specifies the current profile for an object.

Syntax

set the revProfile of object to profileName

Example Code

```
set the revProfile of button "OK" to "French"
set the revProfile of group "Menubar" to "Short Menus"
set the revProfile of me to "Master"
if the revProfile of this stack is "LargeFont" then beep
```

Comments

Use the revProfile property to switch sets of property values on the fly.

Value:

The revProfile of an object is the name of one of the object's profiles.

Comments:

When you change the value of a property for an object, the value is stored in the object's current profile. If you later switch back to that profile, the stored values are restored. For example, if you create a profile named "Fluffy" for a button, then set the button's backgroundColor property to "pink", that setting of the backgroundColor property is stored with the Fluffy profile. If you later set the button's profile to "Fluffy", the pink color is restored.

If the profileName does not exist for the object, setting the profile either causes an error or creates a new profile with that name for the object. You change this setting in the "Property Profiles" pane of the Preferences window.

(The revProfile property is implemented as a custom property, part of the "cRevGeneral" custom property set. For this reason, you can also refer to it using the custom property set notation, as cRevGeneral["profile"].)

Important! The revProfile property is part of the Profile library. To ensure that the property works in a standalone application, in Step 3 of the Distribution Builder window, make sure the "Include profiles and allow switching" option on the Profiles tab is checked.

Note: When included in a standalone application, the Profile library is implemented as a hidden group and made available when the group receives its first openBackground message. During the first part of the application's startup process, before this message is sent, the revProfile property is not yet available. This may affect attempts to use this property in startup, preOpenStack, openStack, or preOpenCard handlers in the main stack. Once the application has finished starting up, the library is available and the revProfile property can be used in any handler.

revPutIntoXMLNode

command

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revAddXMLNode command, revAppendXML command, revSetXMLAttribute command, revStartXMLData message, revXMLChildContents function, revXMLNodeContents function

Summary

Sets the contents of a node in an XML tree.

Syntax

revXMLPutIntoXMLNode treeID,node,newContents

Example Code

```
revPutIntoXMLNode 12,"/Article/Author","Jane Li"  
revPutIntoXMLNode myCurrentNode,dataPaths["current"],field "Data"
```

Comments

Use the revPutIntoXMLNode command to put data into a node, between the node's start and end tags.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The nodePath is the path to the node whose contents you want to set.

The newContents is the text that the node will contain.

Comments:

If the revPutIntoXMLNode command encounters an error, the result is set to an error message beginning with "xmlerr".

Tip: To put Unicode text into a node, first use the uniDecode function to encode the text as UTF-8:

revPutIntoXMLNode the docId of field "xmltree",selectedNode,


```
uniDecode(the unicodeText of field "Contents","UTF8")
```

Important! The `revPutIntoXMLNode` command is part of the XML library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revQueryDatabase

function

Synonyms

revdb_query

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCloseCursor command, revDataFromQuery function, revExecuteSQL command, revOpenDatabase function, revQueryDatabaseBLOB function, About connecting to and using SQL databases, How to create a record set in a SQL database, How to move through the records in a record set (database cursor)

Summary

Selects records in a database according to a SQL query.

Syntax

```
revQueryDatabase(databaseID,SQLQuery[, {variablesList | arrayName}])
```

Example Code

```
revQueryDatabase(2,"SELECT * FROM EmpStats")
revQueryDatabase(currentDB,field "Query")
revQueryDatabase(the database of me,myQuery,"myVar1","myVar2","myVar3")
```

Comments

Use the revQueryDatabase function to select records in a database to work on.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

The SQLQuery is a string in Structured Query Language that contains a SELECT statement. (Do not include a semicolon at the end of the SQLQuery.)

The variablesList consists of one or more variable names (or expressions that evaluate to variable names), separated by commas.

The arrayName is the name of a single array variable whose keys are sequential numbers.

Note: The variable names or arrayName must be enclosed in quotes; otherwise, the variable's value rather than its name is passed to the revQueryDatabase function.

Value:

The revQueryDatabase function returns a record set ID which designates the record set (database cursor) selected by the SQLQuery. The record set ID is an integer.

Comments:

The SQLQuery may contain one or more placeholders, which are sequential numbers prepended by a colon. The revQueryDatabase function substitutes the corresponding variable name in the variablesList for each of these placeholders. For example, if you have two variables called "valueX" and "valueY", you can use a SQLQuery that includes placeholders as follows:

```
get revQueryDatabase(myID,"INSERT INTO emp() VALUES(:1,:2,:1)",  
  
"valueX","valueY")
```

The content of the variable valueX is substituted for the ":1" in the SQLQuery (in both places where ":1" appears), and the content of valueY is substituted for ":2".

If you specify an arrayName rather than a list of ordinary variables, the revExecuteSQL command substitutes the corresponding element of the array for each of the placeholders in the query:

```
get revQueryDatabase(myID,"INSERT INTO emp() VALUES(:1,:2,:1)",  
  
myArray)
```

The content of the element myArray[1] is substituted for the ":1" in the SQLQuery (in both places where ":1" appears), and the content of myArray[2] is substituted for ":2".

To pass binary data in a variable in the variablesList, prepend "*b" to the variable name. The revQueryDatabase function strips the binary marker "*b" and passes it to the database as binary data, rather than text data. To pass binary data in an array element, prepend "*b" to the element's key.

Note: The revQueryDatabase function caches up to 64K of data at a time to avoid repeated server transactions. Because of this, any binary data fields are limited to 64K, and larger fields are truncated. To handle larger amounts of binary data, use the revQueryDatabaseBLOB function instead.

Tip: To execute a SQL query that does not return a record set (database cursor) (such as INSERT or DELETE), use the revExecuteSQL command instead. Since revExecuteSQL does not allocate storage for returned data, it is more efficient than revQueryDatabase for SQL queries where no data is returned.

If the query is not successful, the revQueryDatabase function returns an error message. The error message is never an integer, so you can check whether the query was successful by checking whether the return value is an integer or not.

Important! The revQueryDatabase function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revQueryDatabase synonym was added in version 2.0.

revQueryDatabaseBLOB

function

Synonyms

revdb_queryblob

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCloseCursor command, revDataFromQuery function, revExecuteSQL command, revOpenDatabase function, revQueryDatabase function, About connecting to and using SQL databases, How to create a record set in a SQL database, How to move through the records in a record set (database cursor)

Summary

Selects records in a database according to a SQL query.

Syntax

revQueryDatabaseBLOB(databaseID,SQLQuery[, {variablesList | arrayName}])

Example Code

```
revDatabaseQueryBLOB(the connectID of this card,myQuery)
get revDatabaseQueryBLOB(currentConnect,field "MyQuery","theDataArray")
put revDatabaseQueryBLOB(5,builtQuery) into currentSearchID
```

Comments

Use the revQueryDatabaseBLOB function to select records in a database to work on, where the records may contain large amounts of binary data (over 64K).

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

The SQLQuery is a string in Structured Query Language that contains a SELECT statement. (Do not include a semicolon at the end of the SQLQuery.)

The variablesList consists of one or more variable names (or expressions that evaluate to variable names), separated by commas.

The arrayName is the name of a single array variable whose keys are sequential numbers.

Note: The variable names or arrayName must be enclosed in quotes; otherwise, the variable's value rather than its name is passed to the revQueryDatabaseBLOB function.

Value:

The revQueryDatabaseBLOB function returns a record set ID which designates the record set (database cursor) selected by the SQLQuery. The record set ID is an integer.

Comments:

The SQLQuery may contain one or more placeholders, which are sequential numbers prepended by a colon. The revQueryDatabaseBLOB function substitutes the corresponding variable name in the variablesList for each of these placeholders. For example, if you have two variables called "valueX" and "valueY", you can use a SQLQuery that includes placeholders as follows:

```
get revQueryDatabaseBLOB(myID,"INSERT INTO emp() VALUES(:1,:2,:1)",  
    "valueX","valueY")
```

The content of the variable valueX is substituted for the ":1" in the SQLQuery (in both places where ":1" appears), and the content of valueY is substituted for ":2".

If you specify an arrayName rather than a list of ordinary variables, the revExecuteSQL command substitutes the corresponding element of the array for each of the placeholders in the query:

```
get revQueryDatabaseBLOB(myID,  
    "INSERT INTO emp() VALUES(:1,:2,:1)",myArray)
```

The content of the element myArray[1] is substituted for the ":1" in the SQLQuery (in both places where ":1" appears), and the content of myArray[2] is substituted for ":2".

To pass binary data in a variable in the variablesList, prepend "*b" to the variable name. The revQueryDatabaseBLOB function strips the binary marker "*b" and passes it to the database as binary data, rather than text data. To pass binary data in an array element, prepend "*b" to the element's key.

Tip: To execute a SQL query that does not return a record set (database cursor) (such as INSERT or DELETE), use the revExecuteSQL function instead. Since revExecuteSQL does not allocate storage for returned data, it is more efficient than revQueryDatabaseBLOB for SQL queries where no data is returned.

Tip: To execute a SQL query that will return data, but where the database does not include BLOBs, use the revQueryDatabase function instead. Unlike revQueryDatabase, revQueryDatabaseBLOB does not cache multiple rows during the query, so it is slower than revQueryDatabase.

If the query is not successful, the revQueryDatabaseBLOB function returns an error message. The error message is never an integer, so you can check whether the query was successful by checking whether the return value is an integer or not.

Important! The revQueryDatabaseBLOB function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revQueryDatabaseBLOB synonym was added in version 2.0.

revQueryResult

function

Synonyms

revdb_cursorerr

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revDatabaseConnectResult function, revDatabaseCursors function, result function, About connecting to and using SQL databases, How to create a record set in a SQL database

Summary

Returns the most recent error message associated with a record set (database cursor).

Syntax

revQueryResult(recordSetID)

Example Code

```
revQueryResult(field "Current Set")
if revQueryResult(thisCursor) is not empty then beep 2
```

Comments

Use the revQueryResult function to check for successful completion of SQL commands.

Parameters:

The recordSetID is the number returned by the revQueryDatabase or revQueryDatabaseBLOB function when the record set was created.

Value:

The revQueryResult function returns a string.

Comments:

If there were no errors associated with the specified record set, the revQueryResult function returns empty.

Important! The revQueryResult function is part of the Database library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

Changes to Transcript:

The revQueryResult synonym was added in version 2.0.

revRecordVideo

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revInitializeVideoGrabber command, revPreviewVideo command, revStopRecordingVideo command, revVideoGrabIdle command, How to capture video from a video camera, How to change video grabber settings

Summary

Records video from a video camera to a file.

Syntax

revRecordVideo filePath

Example Code

```
revRecordVideo "/Disk/Folder/file.mov"  
revRecordVideo "mymovie.avi"  
revRecordVideo theFileName
```

Comments

Use the revRecordVideo command to record a movie file.

Parameters:

The filePath is the name and location of the file that will hold the recorded video data. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

Comments:

You must use the revInitializeVideoGrabber command to open the video grabber before you can use the revRecordVideo command.

If you specified "QT" as the video method when you executed the revInitializeVideoGrabber command, the recorded video is stored in QuickTime format. If you specified "VFW", the recorded video is stored in AVI format.

If the video grabber was already recording video to a file, executing the revRecordVideo command again stops that recording and starts a new one.

Important! If you are using QuickTime for video capture, execute the `revVideoGrabIdle` command periodically while previewing or recording video. Not doing so may cause the video in the video grabber to stutter or display strange screen artifacts

To stop recording the video input, use the `revStopRecordingVideo` command.

Important! The `revRecordVideo` command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revRollBackDatabase

command

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCommitDatabase command, revCloseDatabase command, revert command, About connecting to and using SQL databases

Summary

Rolls back recent changes to a database.

Syntax

revRollBackDatabase databaseID

Example Code

```
revRollBackDatabase monthlyExpenses
```

Comments

Use the revRollBackDatabase command to discard changes to a database.

Parameters:

The databaseID is the number returned by the revOpenDatabase function when the database was opened.

Comments:

If the rollback is not successful, the result function returns an error message that begins with the string "revdberr".

Note: Not all databases support the revRollBackDatabase command.

Important! The revRollBackDatabase command is part of the Database library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revRotatePoly

command

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

angle property, flip command, rotate command, select keyword, How to investigate the Revolution custom libraries, Object menu > Rotate

Summary

Rotates a graphic by a specified angle.

Syntax

revRotatePoly graphicReference,angleInDegrees

Example Code

```
revRotatePoly the long ID of graphic "My Curve",45 -- rotates 45 degrees
revRotatePoly "graphic" && it, 10 * theStartAngle
revRotatePoly the name of graphic 3 of stack "Draw", field "Rotate by"
```

Comments

Use the revRotatePoly command to rotate a line, curve, or irregular polygon graphic.

Parameters:

The graphicReference is any expression that evaluates to a reference to a graphic whose style property is set to "line", "curve", or "polygon".

Important! The revRotatePoly command does not accept direct graphic references. For example, the following statement causes an error message:

```
revRotatePoly graphic "My Graphic",45 -- CAN'T USE THIS FORM
```

Instead, use a form that evaluates to a graphic reference, like this:

```
revRotatePoly the name of graphic "My Graphic",90 -- use this form
revRotatePoly ("field" && quote & "My Field" & quote),22 -- or this
```

The angleInDegrees is a number, or an expression that evaluates to a number.

Comments:

If the `angleInDegrees` is positive, the image rotates counterclockwise. If the `angleInDegrees` is negative, the image rotates clockwise.

Important! The `revRotatePoly` command is part of the Common library. To ensure that the command works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The `revRotatePoly` command is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revRotatePoly` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revRotatePoly` command can be used in any handler.

revSetCardProfile

command

Synonyms

Objects

Profile library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revProfile property, revSetStackProfile command, revSetStackFileProfile command, About properties and property profiles, Property Profiles Tutorial, How to change the profile for all the objects in a stack or card

Summary

Changes the current profile used for a card and all its objects.

Syntax

revSetCardProfile profileName[,stackName]

Example Code

```
revSetCardProfile "MetallicLook"  
revSetCardProfile "Master",the mainStack of this stack
```

Comments

Use the revSetCardProfile command to change property settings on a card to a different set of stored settings.

Parameters:

The profileName specifies which profile to use.

The stackName is the short name of an open stack. If you don't specify a stack, the current card of the topStack is changed.

Note: Although the revSetCardProfile command changes a card, you specify a stack name, not a card name.

Comments:

A profile is a set of property settings for an object. You create a profile for an object, and give it a name, using the Property Profiles pane in the object's property inspector. When you change that object to use a profile, the properties you specified in the profile are changed to the settings you specified. This helps

you quickly switch the appearance and behavior of the object without needing to set its properties individually.

The `revSetCardProfile` command changes the profile of the current card of the specified stack and all the objects on that card. If an object does not have a profile with the specified `profileName`, the object's properties are not changed.

Important! The `revSetCardProfile` command is part of the Profile library. To ensure that the command works in a standalone application, in Step 3 of the Distribution Builder window, make sure the "Include profiles and allow switching" option on the Profiles tab is checked.

revSetDatabaseDriverPath

command

Synonyms

Objects

Database library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

defaultFolder property, revOpenDatabase function, specialFolderPath function

Summary

Specifies where the Database library should look for database drivers.

Syntax

revSetDatabaseDriverPath driverFolder

Example Code

```
revSetDatabaseDriverPath "../drivers/"  
revSetDatabaseDriverPath the defaultFolder
```

Comments

Use the revSetDatabaseDriverPath command if you want to place the database drivers your application uses somewhere other than the same folder as the application.

Parameters:

The driverFolder is the name and location of the folder where the database driver files are stored.

Comments:

To use an external SQL database, the Database library needs to communicate via a database driver for the database you're using. By default, the Database library looks for drivers in the same folder as the application. If you try to work with the Database library, and the database driver is not where Revolution expects it to be, a script error will result.

If the database drivers your application needs are installed in a different folder, use the revSetDatabaseDriverPath command before you use any other commands or functions in the Database library, and before you display a field that you've set up to show data from a database (using the Database pane in the field's property inspector). Usually, the best place for this command is in your stack's startup or preOpenStack handler.

Important! The `revSetDatabaseDriverPath` command is part of the Database library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Database Library" option on the Inclusions tab is checked.

revSetSpeechPitch

command

Synonyms

Objects

Speech library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revSetSpeechPitch command, revSetSpeechSpeed command, revSetSpeechVoice command, revSpeak command

Summary

Sets the pitch (whether the voice is high or low) to be used with text to speech.

Syntax

revSetSpeechPitch pitchLevel

Example Code

```
revSetSpeechPitch 100
revSetSpeechPitch (the thumbPosition of scrollbar "Pitch")
```

Comments

Use the revSetSpeechPitch command to change the way speech sounds.

Parameters:

The pitchLevel is an integer between 30 and 127.

Comments:

The greater the pitchLevel, the higher-pitched the computer is when speaking text with the revSpeak command. Use a greater pitchLevel for a treble voice and a lower pitchLevel for a bass voice.

The pitchLevel is logarithmically related to the voice frequency: the range 30 to 127 corresponds approximately to the range from 50Hz to 12,000Hz. If you need to convert from Hertz (cycles per second) to the pitchLevel numbers used with this command, use the following custom function:

```
function pitchFromHertz theHertzFrequency
    get (ln(theHertzFrequency) - ln(261.625))/ln(1.05946309434)
    return round(60 + it)
end pitchFromHertz
```

The pitch specified by the `revSetSpeechPitch` command is used for all speeches following the command during the current session. If the computer is already speaking when you execute the command, the new pitch affects the current speech.

If text to speech is not available on the current system, the `revSetSpeechPitch` command sets the result function to an error message.

Important! The `revSetSpeechPitch` command is part of the Speech library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revSetSpeechSpeed

command

Synonyms

Objects

Speech library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revSetSpeechPitch command, revSetSpeechVoice command, revSpeak command

Summary

Sets the speed at which the revSpeak command speaks.

Syntax

revSetSpeechSpeed wordsPerMinute

Example Code

```
revSetSpeechSpeed 30  
revSetSpeechSpeed field "WPM"
```

Comments

Use the revSetSpeechSpeed command to speed up or slow down speech.

Parameters:

The wordsPerMinute is an integer between 1 and 300.

Comments:

The greater the wordsPerMinute, the faster the speech. (A normal speech rate for English is approximately 150 words per minute.)

The speed specified by the revSetSpeechSpeed command is used for all speeches following the command during the current session. If the computer is already speaking when you execute the command, the new speed affects the current speech.

If text to speech is not available on the current system, the revSetSpeechSpeed command sets the result function to an error message.

Important! The revSetSpeechSpeed command is part of the Speech library. To ensure that the command works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revSetSpeechVoice

command

Synonyms

Objects

Speech library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revSetSpeechPitch command, revSetSpeechSpeed command, revSpeak command, revSpeechVoices function

Summary

Specifies which voice to use with the revSpeak command.

Syntax

revSetSpeechVoice voiceName

Example Code

```
revSetSpeechVoice "Victoria"  
revSetSpeechVoice the storedVoice of this stack  
revSetSpeechVoice any line of revSpeechVoices()
```

Comments

Use the revSetSpeechVoice command to change the way speech sounds.

Parameters:

The voiceName is the name of a voice installed on the system.

Comments:

To get a list of voices installed on the current system, use the revSpeechVoices function.

If the voiceName you specify is not installed, the revSetSpeechVoice command sets the voice to the default voice set by the system.

The voice specified by the revSetSpeechVoice command is used for all speeches following the command during the current session, but if the computer is already speaking when you execute the command, the new voice does not affect the current speech.

If text to speech is not available on the current system, the revSetSpeechVoice command sets the result function to an error message.

If the `revSetSpeechVoice` command is executed while a speech is being spoken, the current speech is halted.

Important! The `revSetSpeechVoice` command is part of the Speech library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revSetStackFileProfile

command

Synonyms

Objects

Profile library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revProfile property, revSetCardProfile command, revSetStackProfile command, Property Profiles Tutorial

Summary

Changes the current profile used for all stacks in a stack file and all their objects.

Syntax

```
revSetStackFileProfile profileName[,stackName]
```

Example Code

```
revSetStackFileProfile "Brights"  
revSetStackFileProfile "SmallControls",myStackName
```

Comments

Use the revSetStackFileProfile command to change property settings throughout a stack file to a different set of stored settings.

Parameters:

The profileName specifies which profile to use.

The stackName is the short name of an open stack. If you don't specify a stack, the stack file that the topStack belongs to is changed.

Note: Although the revSetStackFileProfile command changes a stack file, you specify a stack name, not a file name.

Comments:

A profile is a set of property settings for an object. You create a profile for an object, and give it a name, using the Property Profiles pane in the object's property inspector. When you change that object to use a profile, the properties you specified in the profile are changed to the settings you specified. This helps you quickly switch the appearance and behavior of the object without needing to set its properties individually.

The `revSetStackFileProfile` command changes the current profile of the specified stack, all the stacks in the same stack file, and all the objects in all these stacks. If an object does not have a profile with the specified `profileName`, the object's properties are not changed.

Important! The `revSetStackFileProfile` command is part of the Profile library. To ensure that the command works in a standalone application, in Step 3 of the Distribution Builder window, make sure the "Include profiles and allow switching" option on the Profiles tab is checked.

revSetStackProfile

command

Synonyms

Objects

Profile library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revProfile property, revSetCardProfile command, revSetStackFileProfile command, Property Profiles Tutorial

Summary

Changes the current profile used for a stack and all its objects.

Syntax

```
revSetStackProfile profileName[,stackName]
```

Example Code

```
revSetStackProfile "LargeFonts", "Preferences"  
revSetStackProfile "Pastel"  
revSetStackProfile "Master", the short name of this stack
```

Comments

Use the revSetStackProfile command to change property settings throughout a stack to a different set of stored settings.

Parameters:

The profileName specifies which profile to use.

The stackName is the short name of an open stack. If you don't specify a stack, the topStack is changed.

Comments:

A profile is a set of property settings for an object. You create a profile for an object, and give it a name, using the Property Profiles pane in the object's property inspector. When you change that object to use a profile, the properties you specified in the profile are changed to the settings you specified. This helps you quickly switch the appearance and behavior of the object without needing to set its properties individually.

The `revSetStackProfile` command changes the current profile of the specified stack and all the objects in the stack. If an object does not have a profile with the specified `profileName`, the object's properties are not changed.

Important! The `revSetStackProfile` command is part of the Profile library. To ensure that the command works in a standalone application, in Step 3 of the Distribution Builder window, make sure the "Include profiles and allow switching" option on the Profiles tab is checked.

revSetVideoGrabberRect

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

globalLoc function, revInitializeVideoGrabber command, How to change the size and position of the video grabber window, How to change video grabber settings

Summary

Changes the size and location of the video grabber window.

Syntax

revSetVideoGrabberRect left,top,right,bottom

Example Code

```
revSetVideoGrabberRect 100,300,100,400  
revSetVideoGrabberRect globalRect(the rect of button "Video")
```

Comments

Use the revSetVideoGrabberRect command to move or resize the video grabber window.

Parameters:

The left, right, top, and bottom are integers.

Comments:

You must use the revInitializeVideoGrabber command to open the video grabber before you can use the revSetVideoGrabberRect command.

The rectangle specified by the left, top, right, and bottom is the rectangle of the video grabber window: the top, left, bottom, and right edges of the video grabber, in absolute (screen) coordinates.

Important! The revSetVideoGrabberRect command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revSetVideoGrabSettings

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revInitializeVideoGrabber command, revVideoGrabDialog command, revVideoGrabSettings function, How to change the size and position of the video grabber window, How to change video grabber settings

Summary

Restores video settings saved by the revVideoGrabSettings function.

Syntax

revSetVideoGrabSettings settingsString

Example Code

```
revSetVideoGrabSettings it
revSetVideoGrabSettings the savedVideoSettings of this card
revSetVideoGrabSettings URL "binfile:Data/Video Settings.dat"
```

Comments

Use the revSetVideoGrabSettings command to restore settings for video capture.

Parameters:

The settingsString is a string of binary data in the format returned by the revVideoGrabSettings function.

Comments:

You specify settings for video capture in the video capture dialog box, which is displayed by the revVideoGrabDialog command. You can get the current settings with the revVideoGrabSettings function and restore them later with the revSetVideoGrabSettings command. This allows you to make changes to the video-capture settings under script control.

Important! The revSetVideoGrabSettings command works only for QuickTime video capture.

Important! The revSetVideoGrabSettings command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revSetXMLAttribute

command

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revAppendXML command, revXMLAttribute function, revXMLAttributes function, revXMLAttributeValues function, revXMLMatchingNode function

Summary

Creates an attribute of a node, or sets the value of an existing attribute.

Syntax

revSetXMLAttribute treeID,node,attributeName,newValue

Example Code

```
revSetXMLAttribute 6,"/Vegetable/Tree/Cedar","height","tall"  
revSetXMLAttribute thisTree,currNode,myString,field "Data"
```

Comments

Use the revSetXMLAttribute command to create and change attributes.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The node is the path to the node whose attribute will be set.

The attributeName is a string.

The newValue is a string.

Comments:

If the attributeName is not already an attribute of the node, a new attribute with the specified name and value is created.

If the attribute already exists, its value is set to the newValue.

If the revXMLSetAttribute command encounters an error, the result is set to an error message beginning with "xmlerr".

Tip: To put Unicode text into an attribute, first use the uniDecode function to encode the text as UTF-8:

```
revSetXMLAttribute myTree,the nodeName of me,  
  
    uniDecode(the unicodeText of it,"UTF8")
```

Important! The revXMLSetAttribute command is part of the XML library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revShowPrintDialog

command

Synonyms

Objects

Printing library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

Note: this command works differently across platforms.

See Also:

answer printer command, print command, printPaperSize property, revPrintField command, revPrintText command, How to investigate the Revolution custom libraries, How to print in landscape mode, File menu > Page Setup..., File menu > Print Card..., File menu > Print Field...

Summary

Controls display of the Page Setup (on Mac OS systems) and Print dialog boxes when using the revPrintField or revPrintText commands.

Syntax

revShowPrintDialog showPageSetup,showPrint

Example Code

```
revShowPrintDialog true,true -- shows both dialogs
revShowPrintDialog false,true -- shows only Print dialog
revShowPrintDialog false,the hilite of button "Set Options"
```

Comments

Use the revShowPrintDialog command to let users specify options for printing.

Parameters:

The showPageSetup is an expression that evaluates to true or false.

The showPrint is an expression that evaluates to true or false.

Comments:

Before printing, applications usually display a dialog box where the user can set certain print-related properties. (This dialog is displayed by the operating system, not by Revolution.) Typically, the options in the dialog box include enlargement or reduction, landscape or portrait mode, and paper size, but the exact options are determined by the printer driver and the operating system. If you use the revShowPrintDialog command in a handler before a revPrintField or revPrintText command, these commands display this dialog box.

Cross-platform note: On Mac OS and OS X systems, the `showPageSetup` parameter controls whether the Page Setup dialog box appears, and the `showPrint` parameter controls whether the Print dialog box appears. On Windows systems, the `showPageSetup` parameter controls whether the standard printing options dialog box appears, and the `showPrint` parameter is ignored.

If the user cancels the print dialog box, the next `revPrintField` or `revPrintText` command in the current handler is skipped, and the result is set to "Cancel" after the `revPrintField` or `revPrintText` command is executed.

If you use the `revShowPrintDialog` command outside a handler where a `revPrintField` or `revPrintText` command is executed, the `revShowPrintDialog` command has no effect. The command sets printing options only for the currently executing handler.

Important! The `revShowPrintDialog` command is part of the Printing library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Printing Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Printing library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revShowPrintDialog` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revShowPrintDialog` command can be used in any handler.

revSpeak

command

Synonyms

Objects

Speech library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revIsSpeaking function, revSetSpeechPitch command, revSetSpeechSpeed command, revSetSpeechVoice command, revStopSpeech command, revUnloadSpeech command, How to make the computer speak out loud, How to speak several phrases in succession, Recipe for speaking an alert message

Summary

Speaks text through the computer's speakers.

Syntax

revSpeak phraseToSpeak

Example Code

```
revSpeak "Hello world"  
revSpeak field "Biography Text"
```

Comments

Use the revSpeak command to use the computer's text-to-speech capability.

Parameters:

The phraseToSpeak is a string of any length.

Comments:

The revSpeak command uses the voice, pitch, and speed specified by the revSetSpeechVoice, revSetSpeechPitch, and revSetSpeechSpeed commands. If you haven't used these commands during the current session to specify a voice, pitch, or speed, the system's settings are used.

Note: If you execute the revSpeak command while another speech is being spoken, the first speech is stopped and the second speech begins immediately. To find out whether the computer is already speaking, use the revIsSpeaking function, as in the following example:

```
if revIsSpeaking then answer "Just a moment..."  
else revSpeak it
```

Important! If your application uses text to speech, you should execute the `revUnloadSpeech` command either when your application is finished using text to speech, when the stack that uses speech is closed (in a `closeStack` handler), or when your application quits (in a shutdown handler). This saves memory.

If text to speech is not available on the current system, the `revSpeak` command sets the result function to an error message.

Important! The `revSpeak` command is part of the Speech library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revSpeechVoices

function

Synonyms

Objects

Speech library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revSetSpeechVoice command, revSpeak command

Summary

Returns a list of available voices to use with the revSetSpeechVoice command.

Syntax

revSpeechVoices([voiceGender])

Example Code

```
revSpeechVoices()  
revSpeechVoices("female")  
put revSpeechVoices("neuter") into button "Voices Menu"
```

Comments

Use the revSpeechVoices function to find out what voices can be used to speak text on the current system.

Parameters:

The voiceGender is one of "male", "female", or "neuter". If you don't specify a voiceGender, all voices are returned.

Value:

The revSpeechVoices function returns a list of voice names, one per line.

Comments:

The revSpeechVoices function returns a list of voices installed on the current system. Each voice is either male, female, or neuter, and you can use the optional voiceGender parameter to request one of the genders.

Important! The revSpeechVoices function is part of the Speech library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revStartXMLData

message

Synonyms

Objects

card

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCreateXMLTree function, revCreateXMLTreeFromFile function, revPutIntoXMLNode command, revStartXMLNode message, revStartXMLTree message, revXMLNodeContents function

Summary

Sent to the current card when the revCreateXMLTree or revCreateXMLTreeFromFile function encounters data between tags while parsing an XML document.

Syntax

revStartXMLData elementData

Example Code

```
on revStartXMLData theData -- store data for this node
  if field "Data" is empty then put theData into field "Data"
end revStartXMLData
```

Comments

Handle the revStartXMLData message if you want to build your own subset of an XML document.

Parameters:

The elementData is the text of the XML element currently being parsed.

Comments:

The revCreateXMLTree or revCreateXMLTreeFromFile functions take XML data and parse it. When you call either of these functions, you can specify whether or not to send messages during the parsing operation.

If you have specified that you want the function to send messages, the revStartXMLData message is sent when the function encounters element data. If you have specified that you don't want messages sent, no revStartXMLData messages are sent.

revStartXMLNode

message

Synonyms

Objects

card

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revAddXMLNode command, revCreateXMLTree function, revCreateXMLTreeFromFile function, revDeleteXMLNode command, revEndXMLNode message, revStartXMLData message, revXMLStartTree message

Summary

Sent to the current card when the revCreateXMLTreeFromFile function encounters an opening tag while parsing an XML file.

Syntax

revStartXMLNode nodeAttributes

Example Code

```
on revStartXMLNode theAttributes -- create a new card for this node
  if "publisher",the currPublisher of this stack

    is among the lines of theAttributes then
      create card
      put theAttributes into field "Attributes"
    end if
  end revStartXMLNode
```

Comments

Handle the revStartXMLNode message if you want to build your own subset of an XML document.

Parameters:

The nodeAttributes is a string containing the attributes of the XML element currently being parsed, one attribute per line. Each attribute name is separated from its value by a comma.

Comments:

The revCreateXMLTree or revCreateXMLTreeFromFile functions take XML data and parse it. When you call either of these functions, you can specify whether or not to send messages during the parsing operation.

If you have specified that you want the function to send messages, the revStartXMLNode message is sent when the function encounters the start of a node. If you have specified that you don't want messages sent, no revStartXMLNode messages are sent.

revStartXMLTree

message

Synonyms

Objects

card

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCreateXMLTree function, revCreateXMLTreeFromFile function, revDeleteXMLTree command, revEndXMLTree message, revStartXMLData message, revStartXMLNode message, revXMLRootNode function

Summary

Sent to the current card when the revCreateXMLTreeFromFile function starts parsing an XML document.

Syntax

revStartXMLTree

Example Code

```
on revStartXMLTree -- prepare a stack to hold a data subset
  ask "Get XML data for which publisher?"
  if it is empty then exit to top -- stop parsing
  set the currPublisher of this stack to myPublisher -- store value
  clone stack "Subtree" of this stack -- new stack to hold data subset
end revStartXMLTree
```

Comments

Handle the revStartXMLTree message if you want to build your own subset of an XML document.

Comments:

The revCreateXMLTree or revCreateXMLTreeFromFile functions take XML data and parse it. When you call either of these functions, you can specify whether or not to send messages during the parsing operation.

If you have specified that you want the function to send messages, the revStartXMLTree message is sent when the function begins parsing the XML. If you have specified that you don't want messages sent, no revStartXMLTree messages are sent.

revStopAnimation

command

Synonyms

Objects

Animation library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

play command, revGoToFramePaused command, revPlayAnimation command, stop command, Animation Builder Tutorial, How to investigate the Revolution custom libraries, Why don't animations run in my standalone?, Tools menu > Animation Builder

Summary

Halts a playing animation.

Syntax

revStopAnimation animationName

Example Code

```
revStopAnimation "Flying Circus"  
revStopAnimation (the animName of the target)
```

Comments

Use the revStopAnimation command from within a handler that's triggered by a key frame to stop the current animation from playing.

Parameters:

The animationName is an expression that evaluates to the name of an animation on the current card.

Comments:

You create and name animations using the Animation Builder. To open the Animation Builder, select the object or objects you want to animate, then choose Tools menu Animation Builder. When you designate a key frame for an object in the Animation Builder, you can specify a message to be sent to that object when the key frame is reached.

The animationName must be an animation that was created on the current card of the defaultStack.

Important! The revStopAnimation command is part of the Animation library. To ensure that the command works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "Animation Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Animation library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revStopAnimation` command is not yet available. This may affect attempts to use this command in `startup`, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revStopAnimation` command can be used in any handler.

revStopPreviewingVideo

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revInitializeVideoGrabber command, revPreviewVideo command, revStopRecordingVideo command

Summary

Stops showing input from a video camera in the video grabber window.

Syntax

revStopPreviewingVideo

Example Code

```
revStopPreviewingVideo  
if isShowingVideo then revStopPreviewingVideo
```

Comments

Use the revStopPreviewingVideo command to stop showing video input.

Comments:

When you stop previewing video, the video grabber window shows a blank screen.

Important! The revStopPreviewingVideo command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revStopRecordingVideo

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revInitializeVideoGrabber command, revRecordVideo command, revStopPreviewingVideo command

Summary

Stops a video capture that was started with the revRecordVideo command.

Syntax

revStopRecordingVideo

Example Code

```
revStopRecordingVideo  
if the mouseClick then revStopRecordingVideo
```

Comments

Use the revStopRecordingVideo command to stop recording video input to a file.

Comments:

The revStopRecordingVideo command stops a recording started with the revRecordVideo command. If no recording is in progress, this command has no effect.

Important! The revStopRecordingVideo command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revStopSpeech

command

Synonyms

Objects

Speech library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revIsSpeaking function, revSpeak command, revUnloadSpeech command

Summary

Halts a speech that was started with the revSpeak command.

Syntax

revStopSpeech

Example Code

```
revStopSpeech  
if userCanHaltSpeech then revStopSpeech
```

Comments

Use the revStopSpeech command to let the user interrupt a lengthy speech, or to stop a speech before it's finished.

Comments:

You can use the revIsSpeaking function to determine whether there's currently a speech in progress.

Important! The revStopSpeech command is part of the Speech library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revUnixFromMacPath

function

Synonyms

Objects

Common library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

create folder command, delete file command, read from file command, rename command, revMacFromUnixPath function, write to file command, About filename specifications and file paths, How to investigate the Revolution custom libraries

Summary

Converts a Mac OS-style pathname to a Unix-style pathname.

Syntax

revUnixFromMacPath(macPathname[,convertOSX])

Example Code

```
revUnixFromMacPath("Disk:Folder:") -- returns "/Disk/Folder/"
answer file revUnixFromMacPath(someXFCN())
put revUnixFromMacPath(myValue,true)
```

Comments

Use the revUnixFromMacPath function to convert a Mac OS-style file path to the Revolution file path format (for example, to convert a pathname returned by an XFCN).

Parameters:

The macPathname is a file or folder pathname in the format used by the Mac OS for file paths.

Value:

The revUnixFromMacPath function returns a string with the file path in the format expected by the Mac OS.

The convertOSX is true or false. If you don't specify the convertOSX, if OS X is running, Revolution assumes the macPathname is an OS X-style path to a Mac OS-style path; otherwise, it assumes the macPathname is a Mac OS-style path.

Comments:

The `revUnixFromMacPath` function converts colons (:) to slashes (/), the folder-level delimiter for Unix pathnames. It also adjusts relative paths and absolute paths to be in Unix standard form.

On Mac OS systems, absolute paths always begin with the name of the disk that the file or folder is on. On OS X systems, the startup disk's name does not appear in absolute file paths. Instead, if a file or folder is on the startup disk, the first part of the file path is the top-level folder that the file is in. If a file or folder is on a disk other than the startup disk, its absolute path starts with "Volumes", followed by the disk name.

The OS X path convention is used by Revolution, but the old Mac OS-style path convention is used by certain applications (such as AppleScript), even on OS X systems. If the `convertOSX` is true (or if you don't specify the `convertOSX` and the application is running under OS X), the `revUnixFromMacPath` function assumes that absolute paths are using the OS X convention. If the `convertOSX` is false, the `revUnixFromMacPath` function assumes that absolute paths use the Mac OS convention.

Revolution always uses the Unix pathname standard for cross-platform compatibility. You need to convert the pathname only if you are passing it to another program or external. If you are using only Revolution commands and functions, you do not need to convert the pathname, since Revolution does it for you.

Important! The `revUnixFromMacPath` function is part of the Common library. To ensure that the function works in a standalone application, you must include it when you create your standalone by making sure at least one of the "Library" options on the Inclusions tab in step 3 of the Distribution Builder is checked. The `revUnixFromMacPath` function is included whenever any Revolution custom library is included.

Note: When included in a standalone application, the Common library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revUnixFromMacPath` function is not yet available. This may affect attempts to use this function in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revUnixFromMacPath` function can be used in any handler.

Changes to Transcript:

The `convertOSX` parameter was introduced in version 2.1.1. In previous versions, the `revUnixFromMacPath` function did not attempt to convert between the Mac OS and OS X conventions described above.

revUnloadSpeech

command

Synonyms

Objects

Speech library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

hasMemory function, revSpeak command

Summary

Removes the operating system's text-to-speech software from memory.

Syntax

revUnloadSpeech

Example Code

```
revUnloadSpeech  
if field "Speak Next" is empty then revUnloadSpeech
```

Comments

Use the revUnloadSpeech command to free up memory when you're done using the Speech library.

Comments:

The operating system loads its text-to-speech software into memory when it's needed by any of the commands and functions in the Speech library. The revUnloadSpeech command lets you unload this software, freeing up the memory it uses, when you're done.

If your application uses text to speech, you should execute the revUnloadSpeech command either when your application is finished using text to speech, when the stack that uses speech is closed (in a closeStack handler), or when your application quits (in a shutdown handler).

Important! The revUnloadSpeech command is part of the Speech library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revUpdateGeometry

command

Synonyms

Objects

Geometry library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

lock messages command, resizeStack message, revCacheGeometry command, Geometry Management Tutorial, How to investigate the Revolution custom libraries

Summary

Moves and resizes objects according to their Geometry pane settings.

Syntax

revUpdateGeometry

Example Code

```
revUpdateGeometry  
if the width of this stack > 100 then revUpdateGeometry
```

Comments

The Revolution development environment automatically executes the revUpdateGeometry command when a resizeStack message is sent. Use the revUpdateGeometry command if your stack handles the resizeStack message without passing it.

Comments:

Normally, you do not need to use the revUpdateGeometry command at all, since Revolution automatically updates geometry when a stack is resized. Use it only if both the following conditions are true:

1. Your stack uses the Geometry pane of the property inspector to automate object positioning and sizing when the stack window is resized, and
2. Either a handler in your stack locks messages and then moves or resizes the stack window, or a resizeStack handler in your stack does not pass the resizeStack message at the end of the handler.

If both these conditions are true, execute the revUpdateGeometry command in the handler, after the resizing is performed, to perform Geometry tasks.

Important! The `revUpdateGeometry` command is part of the Geometry library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "Geometry Library" option on the Inclusions tab is checked.

Note: When included in a standalone application, the Geometry library is implemented as a hidden group and made available when the group receives its first `openBackground` message. During the first part of the application's startup process, before this message is sent, the `revUpdateGeometry` command is not yet available. This may affect attempts to use this command in startup, `preOpenStack`, `openStack`, or `preOpenCard` handlers in the main stack. Once the application has finished starting up, the library is available and the `revUpdateGeometry` command can be used in any handler.

revVideoFrameImage

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

imageData property, revPreviewVideo command, revRecordVideo command, How to capture video from a video camera, How to take a picture with a video camera

Summary

Puts the current frame in the video grabber into a variable.

Syntax

revVideoFrameImage frameWidth,frameHeight,dataVariable

Example Code

```
revVideoFrameImage 400,300,"myData"  
revVideoFrameImage the width of image "Snapshot",  
    the height of image "Snapshot","thisFrame"
```

Comments

Use the revVideoFrameImage command to capture a single frame of video.

Parameters:

The frameWidth is a positive integer.

The frameHeight is a positive integer.

The dataVariable is the name of an existing variable.

Comments:

The revVideoFrameImage command returns binary data in the same format as the imageData property of images, placing it in the dataVariable. To show the captured frame in an image, set the image's imageData property to the value placed in the dataVariable, as in the following example:

```
put empty into frameContainer  
revVideoFrameImage 400,300,"frameContainer"  
set the imageData of image "Picture" to frameContainer
```

If the `frameWidth` and `frameHeight` are not the same as the width and height of the video grabber window, the frame is scaled before being returned by the `revVideoFrameImage` function. The data returned by the command always uses the `frameWidth` and `frameHeight`, even if this means the video must be stretched or shrunk to fit. For example, if the video grabber window is 100 x 100 pixels, but you specify 200 as the `frameWidth` and 100 as the `frameHeight`, the returned data fits an image 200 x 100 pixels, with the horizontal dimension stretched to fit.

Important! The `revVideoFrameImage` command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revVideoGrabDialog

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revInitializeVideoGrabber command, revSetVideoGrabSettings command, revVideoGrabSettings function, How to change video grabber settings

Summary

Displays a dialog box for configuring QuickTime or Video for Windows video capture.

Syntax

revVideoGrabDialog [settingsType]

Example Code

```
revVideoGrabDialog  
revVideoGrabDialog "format"  
revVideoGrabDialog the label of button "Dialog Type"
```

Comments

Use the revVideoGrabDialog command to specify settings for use with the video grabber.

Parameters:

The settingsType is one of the following:

compression: Settings for compression when recording video

format: Video format, dimensions, and image depth

display: Appearance of previewed video in the video grabber

source: Video input channels and hue, contrast, and saturation settings

Note: Some video camera drivers don't support some settingsTypes.

Comments:

You must use the revInitializeVideoGrabber command to open the video grabber before you can use the revVideoGrabDialog command.

If you specified "QT" as the video method when you executed the `revInitializeVideoGrabber` command, do not specify a `settingsType`. QuickTime video recording settings are found in a single dialog box, which you display with the following statement:

```
revVideoGrabDialog
```

If you specified "VFW", use the `settingsType` parameter to specify which dialog box you want to show.

Important! The `revVideoGrabDialog` command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revVideoGrabIdle

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

idle message, revInitializeVideoGrabber command, revPreviewVideo command, revRecordVideo command

Summary

Gives processing time to QuickTime during video previewing and recording.

Syntax

2revVideoGrabIdle

Example Code

```
revVideoGrabIdle  
send "revVideoGrabIdle" to me in 10 milliseconds
```

Comments

Use the revVideoGrabIdle command to avoid display problems during video capture.

Comments:

When using QuickTime for video capture, you need to execute the revVideoGrabIdle command periodically in order to let QuickTime update the display. (Not doing so may cause the video in the video grabber to stutter or display strange screen artifacts.) If you have issued the revPreviewVideo or revRecordVideo command, be sure to call the revVideoGrabIdle command periodically until you stop previewing or recording video.

There is no specific interval to use between calls to revVideoGrabIdle. If you are seeing jittering or strange artifacts in the video grabber window, try decreasing the interval and calling revVideoGrabIdle more often.

Note: If you specified "VFW" as the video method when you executed the revInitializeVideoGrabber command, you don't need to call revVideoGrabIdle. If the video capture is using Video for Windows, the revVideoGrabIdle command is ignored and has no effect. You need to call revVideoGrabIdle only if you specified "QT" as the video method.

Important! The `revVideoGrabIdle` command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revVideoGrabSettings

command

Synonyms

Objects

Video library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

revSetVideoGrabSettings command, revVideoGrabDialog command

Summary

Puts the current video capture settings, stored as binary data, into a variable.

Syntax

revVideoGrabSettings dataVariable

Example Code

```
revVideoGrabSettings()  
set the savedVideoSettings of this card to revVideoGrabSettings()  
put revVideoGrabSettings() into URL "binfile:/Data/Video Settings.dat"
```

Comments

Use the revVideoGrabSettings command to store video-capture settings you specify in the video capture dialog box.

Parameters:

The dataVariable is the name of an existing variable.

Comments:

The revVideoFrameImage command returns binary data, placing it in the dataVariable.

You specify settings for video capture in the video capture dialog box, which is displayed by the revVideoGrabDialog command. You can get the current settings with the revVideoGrabSettings command and restore them later with the revSetVideoGrabSettings command. This allows you to make changes to the video-capture settings under script control.

The value returned by the revVideoGrabSettings command consists of binary data and is not human-readable, but you can store it in a file or custom property and restore your settings later using the revSetVideoGrabSettings command.

Important! The `revVideoGrabSettings` command works only for QuickTime video capture. If you are capturing video with Video for Windows, the `revVideoGrabSettings` command does not return a meaningful value.

Important! The `revVideoGrabSettings` command is part of the Video library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "All Other Libraries" option on the Inclusions tab is checked.

revXMLAddDTD

command

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revAppendXML command, revCreateXMLTree function, revCreateXMLTreeFromFile function, revXMLValidateDTD function

Summary

Adds an internal DTD to an existing XML tree.

Syntax

revXMLAddDTD treeID,DTDDText

Example Code

```
revXMLAddDTD 12,URL "file:ExampleDTD.txt"  
revXMLAddDTD theCurrTree,the templatedDTD of me
```

Comments

Use the revXMLAddDTD command to specify the format of an XML tree.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The DTDDText is a string that makes up a valid Document Type Definition.

Comments:

If the revXMLAddDTD command encounters an error, the result is set to an error message beginning with "xmlerr".

Important! The revXMLAddDTD command is part of the XML library. To ensure that the command works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLAttribute

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revSetXMLAttribute command, revXMLAttributes function, revXMLAttributeValues function, revXMLNodeContents function, revXMLMatchingNode function

Summary

Returns the value of the specified attribute of the specified node of an XML tree.

Syntax

revXMLAttribute(treeID,node,attributeName)

Example Code

```
revXMLAttribute(2,"/","Timestamp")
put revXMLAttribute(currTree,currNode,the short name of field x)

    into field x
```

Comments

Use the revXMLAttribute function to get an attribute's value.un

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The node is the path to the node whose attribute value you want to get.

The attributeName is the name of the attribute.

Value:

The revXMLAttribute function returns a string.

Comments:

If the revXMLAttribute function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLAttribute function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLAttributes

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revSetXMLAttribute command, revXMLAttribute function, revXMLAttributeValues function, revXMLChildContents function, revXMLMatchingNode function

Summary

Returns a list of all attributes and their values for the specified node.

Syntax

revXMLAttributes(treeID,node,valueDelim,attributeDelim)

Example Code

```
revXMLAttributes(3,"/Continents"," : ",return)
put revXMLAttributes(myTree,thisNode," - ",";") into field "Attr"
repeat for each line thisLine in revXMLAttributes(1,"/",tab,return)
```

Comments

Use the revXMLAttributes function to display a node's attributes or to scan each attribute in a repeat loop.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The node is the path to the node whose attributes you want to list.

The valueDelim is a string that separates an attribute's value from its name.

The attributeDelim is a string that separates attribute name/value pairs from each other.

Value:

The revXMLAttributes function returns a string.

Comments:

If the revXMLAttributes function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLAttributes function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLAttributeValues

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revSetXMLAttribute command, revXMLAttribute function, revXMLAttributes function, revXMLChildContents function, revXMLMatchingNode function

Summary

Returns a list of the values of a specified attribute for the specified nodes in an XML tree.

Syntax

```
revXMLAttributeValues(treeID,startNode,childName,attributeName,  
  
    delimiter,depth)
```

Example Code

```
revXMLAttributeValues(2,"/",,"Age",return,-1)  
get revXMLAttributeValues(thisTree,thisNode,field "Type",comma,2)
```

Comments

Use the revXMLAttributeValues function to list all the values an attribute has in an XML tree or a section of an XML tree, or to get the range of possible values.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The startNode is the path to the node where you want to start.

The childName is a string specifying which child nodes to scan. If it is empty, all child nodes are scanned. Otherwise, only child nodes whose name matches the childName are scanned.

The attributeName is the name of the attribute to scan for.

The delimiter is a string that separates each value from the rest.

The depth specifies how many generations of the XML tree to scan. If you specify zero, only the startNode is scanned; if you specify 1, the startNode and its child nodes are scanned, but not their child nodes; and so on. To scan all generations, specify -1 as the depth.

Value:

The revXMLAttributeValues function returns a string.

Comments:

If the revXMLAttributeValues function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLAttributeValues function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLChildContents

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revAppendXML command, revXMLAttributes function, revXMLChildNames function, revXMLNodeContents function, revXMLText function, revXMLTree function

Summary

Returns a list of the tags and text contents of the specified nodes.

Syntax

```
revXMLChildContents(treeID,startNode,tagDelim,nodeDelim,  
  
    includeChildCount,depth)
```

Example Code

```
revXMLChildContents(2,"/Animal","/",space,false,-1)  
put revXMLChildContents(myTree,thisNode,tab,return,true,3) after it
```

Comments

Use the revXMLChildContents function to get information about a section of an XML tree.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The startNode is the path to the node where you want to start.

The tagDelim is a string that separates each tag name in a child node from the rest.

The nodeDelim is a string that separates each child node from the rest.

The includeChildCount is true or false. If the includeChildCount is true, each node's name is followed by the number of children it has, in brackets.

The depth specifies how many generations of the XML tree to show. If you specify zero, only the startNode is shown; if you specify 1, the startNode and its child nodes are shown, but not their child nodes; and so on. To show all generations, specify -1 as the depth.

Value:

The revXMLChildContents function returns a string.

Comments:

If the revXMLChildContents function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLChildContents function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLChildNames

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revXMLChildContents function, revXMLFirstChild function

Summary

Returns a list of the child nodes under the specified node in an XML tree.

Syntax

revXMLChildNames(treeID,startNode,nameDelim,childName,includeChildCount)

Example Code

```
revXMLChildNames(1,the currNode of me,comma,,false)
get revXMLChildNames(currTree,line 2 of theNodes,return,"Grass",true)
```

Comments

Use the revXMLChildNames function to find out what child nodes are under a parent node.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The startNode is the path to the node whose child nodes you want to list.

The nameDelim is a string that separates each child node's name from the rest.

The childName is a string specifying which child nodes to list. If it is empty, all child nodes are listed. Otherwise, only child nodes whose name matches the childName are listed.

The includeChildCount is true or false. If the includeChildCount is true, each node's name is followed by the number of children it has, in brackets.

Value:

The revXMLChildNames function returns a string.

Comments:

If the revXMLChildNames function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLChildNames function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLFirstChild

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revXMLChildNames function, revXMLNextSibling function, revXMLParentNode function, revXMLPreviousSibling function

Summary

Returns the path to a node's first child node.

Syntax

revXMLFirstChild(treeID,parentNode)

Example Code

```
revXMLFirstChild(1,"/State/City")  
put revXMLFirstChild(thisTree,thisNode) into thisNode
```

Comments

Use the revXMLFirstChild function to begin scanning the child nodes of a parent node.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The parentNode is the path to the node whose child node you want to find.

Value:

The revXMLFirstChild function returns a string consisting of the path to the first child node.

Comments:

If the revXMLFirstChild function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLFirstChild function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLMatchingNode

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revXMLAttributeValues function, revXMLChildNames function, revXMLTree function

Summary

Finds the node in an XML tree where the specified attribute of the node has the specified value.

Syntax

```
revXMLMatchingNode(treeID,startNode,childName,  
  
    attributeName,attributeValue,depth)
```

Example Code

```
revXMLMatchingNode(1,revXMLRootNode(1),,"PubYear","2001",-1)  
put revXMLMatchingNode(thisTree,it,field "Category",thisAttr,"Yes",4)  
  
    after foundNodePaths
```

Comments

Use the revXMLMatchingNode function to search for a node by its attributes.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The startNode is the path to the node where you want to start.

The childName is a string specifying which child nodes to scan. If it is empty, all child nodes are scanned. Otherwise, only child nodes whose name matches the childName are scanned.

The attributeName is the name of the attribute you want to examine.

The attributeValue is the value that the attributeName must have to be found.

The depth specifies how many generations to scan. If you specify 1, the parentNode's child nodes are scanned, but not their child nodes. To scan all generations, specify -1 as the depth.

Value:

The revXMLMatchingNode function returns a string consisting of the path to the first node containing the specified attribute with the specified value.

Comments:

If the revXMLMatchingNode function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLMatchingNode function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLNextSibling

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revXMLFirstChild function, revXMLParentNode function, revXMLPreviousSibling function, revXMLRootNode function

Summary

Returns the path to a child node's next sibling node.

Syntax

revXMLNextSibling(treeID,siblingNode)

Example Code

```
revXMLNextSibling(3,"/City/Publisher")  
put revXMLNextSibling(the currTree of me,thisNode) into nextNode
```

Comments

Use the revXMLNextSibling function to go forward through the list of nodes on the current level.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The siblingNode is the path to the node where you want to start.

Value:

The revXMLNextSibling function returns a string consisting of the path to the next node with the same parent as the siblingNode. If there is no next sibling node, the revXMLNextSibling function returns empty.

Comments:

If the revXMLNextSibling function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLNextSibling function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLNodeContents

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revAddXMLNode command, revAppendXML command, revPutIntoXMLNode command, revXMLAttribute function, revXMLChildContents function

Summary

Returns the text contained in the specified node in an XML tree.

Syntax

revXMLNodeContents(treeID,node)

Example Code

```
revXMLNodeContents(3,"/Book/Chapter/Section")  
put revXMLNodeContents(thisDoc,mySection) into field "Current Section"
```

Comments

Use the revXMLNodeContents function to get the text contents of a node.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The node is the path to the node whose contents you want to get.

Value:

The revXMLNodeContents function returns a string.

Comments:

If the revXMLNodeContents function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLNodeContents function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLNumberOfChildren

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revXMLChildContents function, revXMLChildNames function, revXMLFirstChild function

Summary

Returns the number of child nodes under the specified node in an XML tree.

Syntax

revXMLNumberOfChildren(treeID,startNode,childName[,depth])

Example Code

```
revXMLNumberOfChildren(3,myNode,, -1)
repeat for revXMLNumberOfChildren(theTree,theNode,"Book",1) times
```

Comments

Use the revXMLNumberOfChildren function to find out how many child nodes a parent node has, or to find out how many of a particular kind of child nodes it has.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The startNode is the path to the node whose child nodes you want to count.

The childName is a string specifying which child nodes to count. If it is empty, all child nodes are counted. Otherwise, only child nodes whose name matches the childName are counted.

The depth specifies how many generations to count. If you specify 0, only the parentNode's direct child nodes are counted, but not their child nodes. If you specify 1, the parentNode's child nodes are counted, along with their child nodes. To count all generations, specify -1 as the depth.

Value:

The revXMLNumberOfChildren function returns a non-negative integer.

Comments:

If the revXMLNumberOfChildren function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLAddElement function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLParent

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revXMLFirstChild function, revXMLNextSibling function, revXMLPreviousSibling function, revXMLRootNode function

Summary

Returns the path to the parent node of a node in an XML tree.

Syntax

revXMLParent(treeID,childNode)

Example Code

```
revXMLParent(1, "/City/Publisher")  
put revXMLParent(thisTree,thisNode) into field "Parent"
```

Comments

Use the revXMLParent function to go up one level in an XML tree.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The childNode is the path to the node where you want to start.

Value:

The revXMLParent function returns a string consisting of the path to the parent node of the childNode. If the childNode is the root node of the XML tree (and therefore has no parent), the revXMLParent function returns empty.

Comments:

If the revXMLParent function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLParent function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In

Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLPreviousSibling function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revXMLFirstChild function, revXMLNextSibling function, revXMLParent function, revXMLRootNode function

Summary

Returns the path to a child node's previous sibling node.

Syntax

revXMLPreviousSibling(treeID,siblingNode)

Example Code

```
revXMLPreviousSibling(22,"/City/Publisher")  
put revXMLPreviousSibling(myDoc,thisNode) into thisNode
```

Comments

Use the revXMLPreviousSibling function to back up through the list of nodes on the current level.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The siblingNode is the path to the node where you want to start.

Value:

The revXMLPreviousSibling function returns a string consisting of the path to the previous node with the same parent as the siblingNode. If there is no previous sibling node, the revXMLPreviousSibling function returns empty.

Comments:

If the revXMLPreviousSibling function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLPreviousSibling function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLRootNode

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revXMLFirstChild function, revXMLNextSibling function, revXMLParent function, revXMLPreviousSibling function

Summary

Returns the path to the starting node of an XML tree.

Syntax

revXMLRootNode(treeID)

Example Code

```
revXMLRootNode(3)
put revXMLRootNode(thisTree) into myStartNode
```

Comments

Use the revXMLRootNode function to get the path to an XML tree's root node.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

Value:

The revXMLRootNode function returns a node path.

Comments:

The root node is the starting node of the XML tree. The corresponding XML element is the one that contains all other elements in the XML document.

If the revXMLRootNode function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLRootNode function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your

standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLText

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCreateXMLTree function, revCreateXMLTreeFromFile function, revDeleteXMLTree command, revXMLAttributes function, revXMLChildContents function, revXMLChildNames function, revXMLNodeContents function, revXMLTree function

Summary

Returns the contents of an XML tree as XML text.

Syntax

revXMLText(treeID[,startNode])

Example Code

```
revXMLText(12)
revXMLText(the xmlID of this card,"/Plants/Trees")
put revXMLText(theID) into URL "file:New Customers.xml"
```

Comments

Use the revXMLText function to turn an XML tree back into an XML document.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The startNode is the path to the node where you want to start. If you don't specify a startNode, the revXMLText function starts at the root node and returns the entire XML tree.

Value:

The revXMLText function returns a non-negative integer.

Comments:

If the revXMLText function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLText function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLTree

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCreateXMLTree function, revCreateXMLTreeFromFile function, revXMLAttributes function, revXMLChildContents function, revXMLChildNames function, revXMLNodeContents function, revXMLText function, revXMLTrees function

Summary

Returns a list of the nodes in an XML tree, in a form that shows their parent/child relationships.

Syntax

revXMLTree(treeID,startNode,nodeDelim,padding,includeChildCount,depth)

Example Code

```
revXMLTree(3, "/Sound", return, space, false, 1)
revXMLTree(currTree, foundNode, space, , true, 2)
put revXMLTree(myTree, theNode, return, tab, false, -1) into field "Node List"
```

Comments

Use the revXMLTree function to obtain a text "tree-style" view of an XML tree's nodes.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The startNode is the path to the node where you want to start.

The nodeDelim is a string that separates each node's name from the rest.

The padding is a string that is placed before the name of each node to show its depth in the tree. No padding is placed before the startNode's name.

The includeChildCount is true or false. If the includeChildCount is true, each node's name is followed by the number of children it has, in brackets.

The depth specifies how many generations of the XML tree to show. If you specify zero, only the startNode is shown; if you specify 1, the startNode and its child nodes are shown, but not their child nodes; and so on. To show all generations, specify -1 as the depth.

Value:

The revXMLTree function returns a string.

Comments:

To display the entire XML tree, specify the root node returned by the revXMLRootNode function as the startNode.

Typically, you use return as the nodeDelimiter and tab as the padding. The resulting list of nodes is displayed one node per line, with as many leading tabs as the node's depth.

If the revXMLTree function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLTree function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLTrees

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCreateXMLTree function, revCreateXMLTreeFromFile function, revDeleteAllXMLTrees command, revDeleteXMLTree command

Summary

Returns a list of all XML trees in memory.

Syntax

revXMLTrees()

Example Code

```
revXMLTrees()  
if myTree is not among the lines of revXMLTrees() then loadTree myTree
```

Comments

Use the revXMLTrees function to find out which XML trees have been created, or when you want to perform an action on all the XML trees you're working with.

Value:

The revXMLTrees function returns a list of tree IDs, one per line.

Comments:

Each tree ID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

If the revXMLTrees function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLTrees function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

revXMLValidateDTD

function

Synonyms

Objects

XML library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

revCreateXMLTree function, revCreateXMLTreeFromFile function, revXMLAddDTD command

Summary

Checks whether the syntax of an XML tree conforms to a DTD.

Syntax

revXMLValidateDTD(treeID,DTDDText)

Example Code

```
revXMLValidateDTD(3,field "DTD")  
put revXMLValidateDTD(field "Tree",myDTD) into field "Errors"
```

Comments

Use the revXMLValidateDTD function to validate an XML tree against a DTD.

Parameters:

The treeID is the number returned by the revCreateXMLTree or revCreateXMLTreeFromFile function when you created the XML tree.

The DTDDText is a Document Type Definition.

Value:

The revXMLValidateDTD function returns empty if the XML tree validates against the DTD.

Comments:

If the revXMLValidateDTD function encounters an error, it returns an error message starting with "xmlerr".

Important! The revXMLValidateDTD function is part of the XML library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the "XML Library" option on the Inclusions tab is checked.

right
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

altKey function, commandKey function, controlKey function, down constant, left constant, mouse function, optionKey function, right constant, shiftKey function

Summary

Equivalent to the string "right".

Syntax

Example Code

```
arrowKey right  
if theKey is right then go next card
```

Comments

Use the right constant to indicate the right arrow key in an arrowKey handler.

right property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomRight property, left property, rectangle property, rightMargin property, topRight property, width property, Object menu > Align Selected Controls > Right, Shortcut to align control edges

Summary

Specifies how far an object's right edge is from the left edge of the window or screen.

Syntax

set the right of object to pixels

Example Code

```
set the right of last field to 22  
set the right of me to item 1 of the clickLoc
```

Comments

Use the right property to change the horizontal placement of a control or window.

Value:

The right of an object is an integer. A negative integer indicates that the position is to the left of the left edge of the screen or card (and therefore cannot be seen).

A stack's right is the distance in pixels from the left edge of the screen to the right edge of the stack window.

A card's or control's right is the distance in pixels from the left edge of the card to the right edge of the card or control.

Comments:

The right of a stack is in absolute (screen) coordinates. The right of a card is always the width of the stack window; setting the right of a card does not cause a script error, but it has no effect. The right of a group or control is in relative (window) coordinates.

Changing the right of an object shifts it to the new position without resizing it. To change an object's width, set its width or rectangle property.

The width property of an object is equal to its right minus its left.

rightMargin

property

Synonyms

Objects

button, field, group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomMargin property, formattedWidth property, leftMargin property, margins property, right property, topMargin property, width property

Summary

Specifies how close text within an object can come to the object's right edge, and how close objects in a group can come to the group's right edge.

Syntax

set the rightMargin of {button | field | group} to pixels

Example Code

```
set the rightMargin of group "Options" to 20
```

Comments

Use the rightMargin property to change the amount of blank space between an object's right edge and its contents.

Value:

The rightMargin is a non-negative integer.

By default, the rightMargin of a newly created field is 8. If the field's wideMargins property is true, the field's rightMargin is set to 14. The default rightMargin setting for a button or group is 4.

Comments:

The rightMargin property of a field or button specifies how many blank pixels are left between the object's right edge and the right edge of its text. The rightMargin of a group specifies how far the group's right edge extends below its rightmost object.

An object's rightMargin property is equal to item 3 of its margins property.

rotate

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

angle property, choose command, flip command, revRotatePoly command, select keyword, Object menu
> Rotate

Summary

Rotates an image through the specified angle.

Syntax

rotate [image] by numberOfDegrees

Example Code

```
rotate by 90  
rotate image "Spinner" by -10
```

Comments

Use the rotate command to turn an image.

Parameters:

The image is any image reference.

The numberOfDegrees is a number, or an expression that evaluates to a number.

Comments:

The entire content of the specified image is selected with the Select paint tool, then rotated. If you don't specify an image, the portion currently selected with the Select paint tool is rotated.

If the numberOfDegrees is positive, the image rotates counterclockwise. If the numberOfDegrees is negative, the image rotates clockwise.

If the rotated image extends beyond the boundaries of the original image, and the image's lockLocation property is set to false, the image expands to accommodate the rotated contents. If the lockLocation is true, the rotated image is cropped to fit within the image's rectangle.

The rotation operation does not have an inverse. Since some amount of distortion is unavoidable when rotating an image by an amount that is not a multiple of 90 degrees, rotating an image clockwise, then counterclockwise by the same amount does not completely restore the original image.

To rotate an image in a reversible way, or to do repeated rotations without progressive distortion of the image, set its angle property instead of using the rotate command. Unlike the rotate command, the angle property affects only the screen display of the image, not the actual picture data in it, so setting it repeatedly does not introduce distortion.

Important! The rotate command cannot be used on a referenced image. Doing so will cause an execution error. To turn a referenced image, set the image's angle property instead.

round

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

average function, numberFormat property, statRound function, trunc function, Recipe for an approximate-equality function

Summary

Rounds off a number to the nearest integer.

Syntax

the round of number
`round(number,precision)`

Example Code

```
the round of 26.2 -- yields 26  
the round of 2.5 -- yields 3 (rounds up)  
round(845.6604,2) -- yields 845.66  
round(9734.22,-3) -- yields 10000
```

Comments

Use the round function to round off numbers and eliminate insignificant digits for financial applications.

Parameters:

The number is any number or expression that evaluates to a number.

The precision is an integer giving the decimal place to round off to.

Value:

The round function returns a number.

Comments:

The round function performs financial-style rounding. If the number is exactly halfway between two numbers, round always rounds the number up if positive, down if negative. (To round off numbers without introducing any statistical upward bias, use the statRound function instead.)

A positive precision indicates a place to the right of the decimal point, and a negative precision indicates a place to the left. For example, 1 rounds off to the nearest tenth, 2 rounds off to the nearest hundredth, -1 rounds off by ten, and so on. If you don't specify a precision, zero is used, meaning that the number is rounded off to a whole number.

Note: The statRound function is equivalent to HyperTalk's "round" function.

roundEnds

property

Synonyms

Objects

graphic, global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

dashes property, lineSize property, roundRadius property, style property

Summary

Specifies whether lines have rounded ends or squared-off ends.

Syntax

set the roundEnds [of graphic] to {true|false}

Example Code

```
set the roundEnds to true  
set the roundEnds of graphic "Connector" to false
```

Comments

Use the roundEnds property to control the appearance of lines on Unix or Windows NT systems.

Value:

The roundEnds of a graphic is true or false.

By default, the roundEnds of a newly created graphic is set to false.

Comments:

If the roundEnds of a graphic is true, the ends of lines are rounded, and so is the intersection of lines within the graphic. If the roundEnds is false, the ends of lines are squared off.

If the graphic's lineSize property is 1, the setting of the roundEnds has no effect.

If the graphic's style property is oval, roundRect, or curve, or if the graphic's lineSize property is less than 2, the setting of this property has no effect.

The global setting of the roundEnds property controls the appearance of lines drawn with the line, rectangle, polygon, or regular paint tools. Once a paint line or shape is drawn, its appearance cannot be changed by changing the global roundEnds property.

Cross-platform note: The setting of this property has no effect on Mac OS or Windows 95/98 systems. On Mac OS systems, lines always have square ends. On Windows 95/98 systems, lines always have rounded ends.

roundRadius

property

Synonyms

roundHeight, roundWidth

Objects

global, graphic

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

dashes property, roundEnds property, roundRect keyword, style property, tool property

Summary

Specifies how round the corners of a round rectangle are.

Syntax

set the roundRadius [of graphic] to number

Example Code

```
set the roundRadius to 1
```

Comments

Use the roundRadius property to control the appearance of rounded rectangles.

Value:

The roundRadius of a graphic is a non-negative integer.

By default, the roundRadius of a newly created graphic is set to 15.

Comments:

The roundRadius determines the diameter of the circle used to draw the corners of a round rectangle. For example, if a round rectangle's roundRadius is 15, each corner of the rectangle is an arc of a circle whose diameter is 15 pixels.

If the graphic's style property is not "roundRect", the setting of this property has no effect.

The global setting of the roundRadius property controls the appearance of round rectangles drawn with the roundRect paint tool. Once a paint rounded rectangle is drawn, its appearance cannot be changed by changing the global roundRadius property.

roundRect

keyword

Synonyms

round rect, round rectangle, roundRectangle

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

brushColor property, brushPattern property, choose command, lineSize property, penColor property, tool function, Object menu > New Control > Round Rect Graphic

Summary

Specifies that a button or graphic is shaped like a rectangle with rounded corners. It also designates the paint tool used to draw rounded-corner rectangles.

Syntax

Example Code

```
set the style of graphic "Outline" to roundRect
choose round rectangle tool
```

Comments

Use the roundRect keyword to paint a rounded rectangle or square with the penColor or to change a graphic or button to a rounded rectangle shape.

Comments:

When using the Round Rectangle tool, the cursor is a crosshairs shape (over images) or an arrow (anywhere else). This tool is used to draw a rounded rectangle or square in the penColor, filled with the brushColor (or brushPattern).

If you try to paint with the Round Rectangle tool on a card that has no images, an image the size of the card is created to hold the painting. If the current card already has one or more images, painting outside the image has no effect.

Setting the style of a button or graphic to roundRect makes the graphic into a rounded rectangle. Rounded rectangle graphics, unlike painted rounded rectangles, can be changed and reshaped: use the height and width properties to change the shape and size.

Important! You can use the synonyms `round rect`, `round rectangle`, and `roundRectangle` to choose the round rectangle paint tool, but not to set a graphic's style property.

RTFText

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

charToNum function, foregroundColor property, htmlText property, numToChar function, text property, textFont property, textSize property, textStyle property, unicodeText property, How to copy styled text between fields, How to export text from a field in RTF format, How to import an RTF file, How to import and export styled text

Summary

Specifies the contents of a field, with its text formatting represented in RTF format.

Syntax

set the RTFText of [chunk of] field to RTFString

Example Code

```
set the RTFText of field "Stuff" to URL "file:New Stuff.rtf"  
put the RTFText of field "Destination" into URL myURL
```

Comments

Use the RTFText property to import and export text in RTF format.

Value:

The RTFText of a field is a string.

Comments:

The RTFText property is a representation of the styled text of the field in RTF format.

Setting the RTFText of a field (or a chunk of a field) sets both the text contents and the font, size, style, and color attributes corresponding to the information in the RTFString. Any other formatting controls in the RTFString are ignored.

The RTFText property interprets the following RTF formatting controls:

Colors: ,

save

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clone command, create folder command, mainStack property, revert command, saveStackRequest message, stackFileType property, substacks property, How to create a file, How to delete a file, How to store preferences or data for a standalone application, Why can't I find a stack I just saved?, Why can't I save a stack?, File menu > Save

Summary

Saves a stack file on the user's system.

Syntax

save stack [as filePath]

Example Code

```
save stack "Targets"  
save this stack as "Backup"  
save stack "Treats" as "/Disk/Folder/File"
```

Comments

Use the save command to save changes to a stack or to save a copy of a stack into another file.

Parameters:

The stack is any open stack.

The filePath specifies the name and location of the file you want to save to. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder. If the file does not exist, Revolution creates it.

Comments:

The save command saves all stacks stored in the same file as the specified stack. That is, if you save a main stack, all substacks of that stack are also saved in the same file, and if you save a substack, its main stack and any other substacks are also saved.

If the stack has not yet been saved and doesn't have a filename, you must use the save...as filePath form.

You cannot save to a standalone application's file; standalones are read-only.

If the stack cannot be saved (for example, if you try to save it to a nonexistent drive or to a CD-ROM), the result function is set to "Can't open stack file".

saveCompressed property

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

compact command, save command

Summary

The saveCompressed property is not implemented and is reserved.

Syntax

Example Code

Comments

saveStackRequest

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeStack message, filename property

Summary

Sent to the current card when the stack is about to be saved.

Syntax

saveStackRequest

Example Code

```
on saveStackRequest -- delete all cards but the first
  repeat with x = the number of cards down to 2
    delete card x
  end if
  pass saveStackRequest -- or the stack won't be saved
end saveStackRequest
```

Comments

Handle the saveStackRequest message if you need to do cleanup or other actions before a stack is saved. For example, if the stack creates temporary objects that should not be saved with the stack, you can write a saveStackRequest handler to delete those objects before the stack is saved.

Comments:

If the stack has not yet been saved, the standard Save dialog box appears. The saveStackRequest message is sent after the user clicks Save in this dialog box, and before the file is saved.

The save action is triggered by the saveStackRequest message. This means that trapping the saveStackRequest message and not passing it prevents the stack from being saved.

scale

property

Synonyms

Objects

EPS, videoClip

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

height property, scaleIndependently property, width property, xScale property, yScale property

Summary

Specifies the size at which an EPS object or video clip is displayed.

Syntax

set the scale of {EPSObject | videoClip} to ratio

Example Code

```
set the scale of videoClip "Splash" to 0.5 -- half size
```

Comments

Use the scale property to change the appearance of EPS objects or video clips.

Value:

The scale of an EPS object or video clip is a positive number.

By default, the scale property of newly created objects is set to 1.

Comments:

The scale is the natural size of the object's content divided by the size of the rectangle the content is placed in. For example, if the ratio is 1, a video clip plays at the movie's normal size; if the ratio is 2, the video clip is blown up by a factor of 2.

For EPS objects, this property is supported only on Unix systems with Display PostScript installed.

scaleIndependently

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

boundingBox property, xScale property, yScale property

Summary

Specifies whether an EPS object's vertical and horizontal dimensions are sized independently or proportionally.

Syntax

set the scaleIndependently of EPSObject to {true | false}

Example Code

```
set the scaleIndependently of EPS 2 to false
```

Comments

Use the scaleIndependently property to control the screen appearance of an EPS object.

Value:

The scaleIndependently of an EPS object is true or false.

Comments:

If the scaleIndependently is true, the PostScript image is scaled vertically to fit the full height of the EPS object and horizontally to fit the full width, even if this causes the image to be distorted.

If the scaleIndependently is false, the PostScript image is scaled proportionally to fit at least one dimension, even if the other dimension does not completely fill the EPS object's rectangle.

This property is supported only on Unix systems with Display PostScript installed.

screenColors

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

colorMap property, colorWorld property, lockColorMap property, screenDepth function, screenType function

Summary

Returns the number of colors the screen can display.

Syntax

the screenColors

screenColors()

Example Code

```
the screenColors
```

```
if the screenColors = 2 then put "black & white" into graphicsLevel
```

Comments

Use the screenColors function to determine the color capacity of the screen. For example, you might display different images depending on the number of colors available.

Value:

The screenColors function returns a positive integer which is a power of two.

Comments:

If the system has more than one monitor, the screenColors function returns the number of colors for the main screen.

The expression

the screenColors

is equal to

the screenDepth^2

The value returned by the `screenColors` function is updated only when you start up the application. If you change the screen settings after starting up the application, you must quit and restart to update the `screenColors`.

screenDepth

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

colorWorld property, exp2 function, lockColorMap property, screenColors function

Summary

Returns the bit depth of the screen.

Syntax

the screenDepth
screenDepth()

Example Code

```
the screenDepth  
if the screenDepth > 8 then show image "Photograph"
```

Comments

Use the screenDepth function to determine the color capacity of the screen. For example, you might display different images depending on the number of colors available.

Value:

The screenDepth function returns 1, 2, 4, 8, 16, 24, or 32.

Comments:

If the system has more than one monitor, the screenDepth function returns the bit depth of the main screen.

The expression

the screenDepth^2
is equal to
the screenColors

The value returned by the screenDepth function is updated only when you start up the application. If you change the screen settings after starting up the application, you must quit and restart to update the screenDepth.

screenGamma

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

dontDither property, import command, screenColors function, screenDepth function

Summary

Specifies the gamma value used for displaying PNG images.

Syntax

set the screenGamma to gammaValue

Example Code

```
set the screenGamma to 2.2
```

Comments

Use the screenGamma property to control the color display of PNG images.

Value:

The screenGamma is a number.

Comments:

PNG images can contain a preferred gamma setting. To display a PNG to best advantage, you may need to adjust the screenGamma property to match what the PNG expects.

Changes in the screenGamma do not apply to PNG images that are already displayed. A PNG image uses the screenGamma that was in effect when it was being decompressed for display.

Tip: To force Revolution to re-decompress a PNG image without leaving the card, put the image into itself:

```
set the screenGamma to 2.2
put image "Test PNG" into image "Test PNG"
```

screenLoc

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

location property, screenRect function, Recipe for centering a stack window on the screen

Summary

Returns the location of the screen's center.

Syntax

the screenLoc

screenLoc()

Example Code

```
the screenLoc
set the loc of window "Dialog" to the screenLoc
```

Comments

Use the screenLoc function to determine the height and width of the screen, or to compute the location of windows relative to the center of the screen.

Value:

The screenLoc function returns two integers separated by a comma.

Comments:

The first item in the value returned by the screenLoc is the distance from the left edge to the center of the screen, in pixels. The second item is the distance from the top edge to the center of the screen, in pixels.

If the system has more than one monitor, the screenLoc function returns the rectangle of the main screen.

screenMouseLoc

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

globalLoc function, mouseLoc function, screenRect function

Summary

Specifies the position of the mouse pointer relative to the screen.

Syntax

set the screenMouseLoc to horizontal,vertical

Example Code

```
put the localLoc(the screenMouseLoc) into localMouse
set the screenMouseLoc to the topLeft of this stack
```

Comments

Use the screenMouseLoc property to find out where the mouse pointer is, or to change the location of the mouse pointer.

Value:

The screenMouseLoc consists of two integers, separated by a comma.

Value:

The screenMouseLoc property reports the position of the mouse pointer in absolute coordinates—that is, relative to the top left of the main screen. (The mouseLoc function is similar, but its coordinates are relative to the defaultStack window rather than the screen.)

If you set this property, the mouse pointer is moved to the specified location on the screen.

Important! Taking over the position of the mouse pointer can disorient and confuse users, and user-interface standards warn against doing so for that reason. Consider setting the screenMouseLoc property only in unusual situations (such as for a kiosk application or game). In normal desktop applications, the user expects a consistent user interface without such surprises. You must use this power only for good.

Changes to Transcript:

Support for using the screenMouseLoc with OS X systems was added in version 2.0.

screenName

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

open process command, screenVendor function, shell function

Summary

Returns the name of the current screen.

Syntax

the screenName

screenName()

Example Code

```
the screenName  
put "myApp -d" && the screenName into commandLine
```

Comments

Use the screenName function to control where applications that Revolution starts up should appear.

Value:

The screenName function returns a string.

Comments:

On Unix systems, the screenName function returns the string provided by the XDisplayName function call. When starting a process with the open process command or the shell function, if you want the process to appear on the same screen as Revolution, use the value returned by the screenName as the argument to the -d option.

This function does not return a useful value on Mac OS and Windows systems.

screenNoPixmaps

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

alwaysBuffer property, destroyWindow property, lockScreen property, pixmapID property, screenSharedMemory property, windowID property

Summary

Specifies whether Revolution creates a data structure to hold the screen image of each window.

Syntax

set the screenNoPixmaps to {true | false}

Example Code

```
set the screenNoPixmaps to true
```

Comments

Use the screenNoPixmaps property to trade off lower memory usage for possible screen flashing.

Value:

The screenNoPixmaps is true or false.

By default, the screenNoPixmaps property is set to false

Comments:

If the screenNoPixmaps property is set to false, Revolution creates a pixmap data structure for each of its windows. Changes to the appearance of the window are created in this offscreen image, then drawn to the actual window all at once. This minimizes screen flashing when objects are redrawn.

If the screenNoPixmaps property is true, no offscreen images are created, and drawing is done into the actual window. This reduces the memory needed for the window, but can result in screen flashing.

If the application is started from a Unix command line, this property can be set to true on startup by using the -nopixmaps option.

screenRect

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

globalLoc function, rectangle property, screenLoc function, Recipe for checking whether the screen is large enough

Summary

Returns the rectangle of the main screen.

Syntax

the screenRect
screenRect()

Example Code

```
the screenRect  
set the rect of this stack to the screenRect
```

Comments

Use the screenRect function to find out the screen size and to scale windows to the size of the screen.

Value:

The screenRect function returns four integers, separated by commas.

Comments:

The first two items in the value returned by the screenRect are always zero. The third item is the width of the screen in pixels. The fourth item is the height of the screen in pixels.

If the system has more than one monitor, the screenRect function returns the rectangle of the main screen.

The value returned by the screenRect function is updated only when you start up the application. If you change the screen settings after starting up the application, you must quit and restart to update the screenRect.

screenSharedMemory

property

Synonyms

screenVCSharedMemory

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

alwaysBuffer property, bufferHiddenImages property, screenNoPmaps property

Summary

Specifies whether images are drawn in shared memory on Unix systems.

Syntax

set the screenSharedMemory to {true | false}

Example Code

```
set the screenSharedMemory to false
```

Comments

Use the screenSharedMemory property to improve drawing speed.

Value:

The screenSharedMemory is true or false.

By default, the sharedScreenMemory is set to true on Unix systems, false on Windows and Mac OS systems.

Comments:

If the screenSharedMemory property is set to true, the engine uses the system's shared memory extension (if available) to draw images. This speeds up image rendering.

Shared memory is not available on all Unix systems, and cannot be used if the application is running across the network rather than locally.

If the application is started from a Unix command line, this property can be set to false on startup by using the -s option.

The setting of this property has no effect on Mac OS or Windows systems.

screenType

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

privateColors property, screenColors function, screenDepth function, screenRect function

Summary

Returns the color capability of the screen.

Syntax

the screenType
screenType()

Example Code

```
the screenType  
if the screenType is "PseudoColor" then useLineImages else usePhotos
```

Comments

Use the screenType function to determine what kind of color or grayscale values the current screen can use.

Value:

The screenType function returns one of the following types:

- StaticGray: a fixed list of shades of gray
- StaticColor: a fixed list of colors
- GrayScale: a list of shades of gray, each changeable by the engine
- PseudoColor: a list of 256 or fewer colors, each changeable by the engine
- TrueColor: a list of 2^{24} (16,777,216) colors
- DirectColor: an unlimited number of colors

Comments:

If the screenType is `StaticGray` or `StaticColor`, the available colors are predefined in a color table. Any colors displayed by Revolution are changed to the closest available color.

If the `screenType` is `GrayScale`, `PseudoColor`, or `TrueColor`, a predefined number of color slots is available, and the engine can change any of those colors to one of a larger set of available colors. 8-bit color displays (256 colors) are usually `PseudoColor`.

If the `screenType` is `DirectColor`, any pixel on the screen can be set to any color that the screen supports.

On some Unix systems, you can change the `screenType` with the `-v` command-line option. See the `xdpyinfo` command and the Unix documentation for more information on visual types and changing the visual type.

If the system has more than one monitor, the `screenType` function returns the color capability of the main screen.

The value returned by the `screenType` function is updated only when you start up the application. If you change the screen settings after starting up the application, you must quit and restart to update the `screenType`.

screenVendor

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

machine function, platform function

Summary

Returns version information about the producer of the windowing software.

Syntax

the screenVendor

screenVendor()

Example Code

```
the screenVendor
```

```
if the screenVendor is originalVendor then pass mouseUp
```

Comments

Use the screenVendor function to get information about the XWindows software version.

Value:

The screenVendor function returns a string.

Comments:

The return value usually includes the company or organization that produced the screen driver, along with version information.

script property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

-- keyword, backScripts function, do command, edit command, editScript message, frontScripts function, function control structure, getProp control structure, licensed function, on control structure, scriptLimits function, setProp control structure, stacksInUse property, How to change a script at run time, How to edit an objectÆs script, Object menu > Object Script, Object menu > Card Script, Object menu > Stack Script, Shortcut to edit the stack script, Shortcut to edit the card script, Shortcut to edit a controlÆs script, Shortcut to edit the selected controlÆs script

Summary

Specifies the contents of an objectÆs script.

Syntax

set the script of object to string

Example Code

```
set the script of button 2 to empty -- clear out the script
set the script of field "Get It" to field "Get It Script"
```

Comments

Use the script property to examine or change an objectÆs script from inside a Transcript statement, instead of using the script editor.

Value:

The script of an object is a string consisting of handlers and comments.

By default, the script of a newly created object is empty.

Comments:

An objectÆs script is the code that controls that object. All of an objectÆs handlers are part of its script property.

You can view and change a script directly by selecting the object and choosing Object menu Object Script. Use the script property within a handler to check the contents of a script, or change the script.

Note: When using a standalone application, an object's script property may not be set to a string containing more than ten statements. This limit is set by line 1 of the scriptLimits function. (This does not limit scripts that are already written: standalone applications can run scripts of any length. However, if the standalone attempts to change an object's script property, and the script contains more than the allowable number of statements, the attempt to set the script causes an error.)

Revolution compiles the script immediately after it is set. This means that it is not possible to write a self-modifying handler, because the currently-executing handler would have to be changed and re-compiled while it was running.

scriptError

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

executionError property, scriptParsingError message

Summary

Removed in version 1.1.

Syntax

get the scriptError

set the scriptError to empty

Example Code

Comments

The scriptError property reported information about the most recent compile error.

For the information formerly reported by the scriptError property, see the scriptParsingError message.

scriptLimits

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

; keyword, keyword, edit command, environment function, insert script command, licensed function, script property, start using command

Summary

Returns the limits on script length for scripts that are edited in a standalone application.

Syntax

the scriptLimits

scriptLimits()

Example Code

```
the scriptLimits  
if the scriptLimits is not empty then answer "Not licensed."
```

Comments

Use the scriptLimits function to determine the limits on changing scripts when using standalone applications created with Revolution.

Value:

The scriptLimits function returns a comma-separated list of four positive integers if the current application is a standalone created with Revolution. If the current application is the development environment, the scriptLimits function returns empty.

Comments:

The four numbers returned by the scriptLimits function are:

- ò the number of statements permitted when changing a script (normally 10)
- ò the number of statements permitted in a do command (normally 10)
- ò the number of stacks permitted in the stacksInUse (normally 50)
- ò the number of objects permitted in the frontScripts and backScripts (normally 10)

Important! The first number does not limit scripts that are already written. In other words, standalone applications can run scripts of any length. However, if the standalone attempts to change an object's script property, and the script contains more than the allowable number of statements, the attempt to set the script causes an error.

The above limits apply only when running standalone applications you create with Revolution. If you are using the development environment, these limits do not apply and the `scriptLimits` function returns empty.

scriptParsingError

message

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

errorDialog message, lockErrorDialogs property

Summary

Sent to an object when its script cannot be compiled.

Syntax

scriptParsingError errorContents

Example Code

Comments

Handle the scriptParsingError message to prevent the standard error window from appearing, when you want to handle the error in a custom handler.

Parameters:

The errorContents specifies the details of the problem that caused the scriptParsingError message to be sent.

Comments:

The scriptParsingError message is sent when a compile error occurs. If the script contains an execution error, the errorDialog message is sent instead when the handler containing the error attempts to run.

A list of possible compile errors is contained in the cScriptErrorsö property of the first card of the stack örevErrorDisplayö. You can view the list with the following statement:

```
answer the cScriptErrors of card 1 of stack "revErrorDisplay"
```

Note: The exact format of the errorContents may change from release to release.

Changes to Transcript:

The `errorContents` parameter was introduced in version 1.1. In previous versions, this information was stored in the `scriptError` property.

scriptTextFont

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

(Platform-dependent)

Summary

Specifies the font face used in the script editor.

Syntax

set the scriptTextFont to fontName

Example Code

```
set the scriptTextFont to "Andale Mono"  
set the scriptTextFont to the textFont of field "Template"
```

Comments

Use the scriptTextFont property to change the appearance of text in the script editor.

Value:

The scriptTextFont is a string.

By default, the scriptTextFont property is set to ôCourierö (on Mac OS and Unix systems) or ôCourier Newö (on Windows systems).

Comments:

On Mac OS systems, if the specified font isnÆt available, the system font (which is set in the Appearance control panel and specifies the font used for menus) is used. On Unix systems, if the specified font isnÆt available, Helvetica is used. On Windows systems, if the specified font isnÆt available, the current Windows font is used.

scriptTextSize

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

(Platform-dependent)

Summary

Specifies the point size of text used in the script editor.

Syntax

set the scriptTextSize to pointSize

Example Code

```
set the scriptTextSize to 14
set the scriptTextSize to (the scriptTextSize - 2)
```

Comments

Use the scriptTextSize property to change the appearance of text in the script editor.

Value:

The scriptTextSize is an integer greater than 4.

By default, the scriptTextSize property is set to 12.

scrollbar

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

button keyword, choose command, field keyword, graphic keyword, image keyword, player keyword, style property, templateScrollbar keyword, tool function

Summary

Designates the Scrollbar tool, which is used to create new scrollbars.

Syntax

Example Code

```
choose scrollbar tool
```

Comments

Use the scrollbar keyword to create scrollbars, progress bars, and drag bars.

Comments:

When using the Scrollbar tool, the cursor is a crosshairs. This tool is used for creating scrollbars by clicking at one corner of the scrollbar and dragging to the opposite corner.

You can set the style of the templateScrollbar to the scrollbar type you want, then draw the scrollbar with the Scrollbar tool. The scrollbar icon on the Tools palette works this way: each choice in the popup menu sets the style of the templateScrollbar, then chooses the Scrollbar tool so you can create the scrollbar.

scrollbar

object

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

templateScrollbar keyword, About object types and object references, Object menu > New Control > Horizontal Scrollbar, Object menu > New Control > Vertical Scrollbar, Object menu > New Control > Scale Bar, Object menu > New Control > Progress Bar

Summary

A control that indicates a position or setting.

Syntax

Example Code

```
set the style of scrollbar "Progress" to progress
if the thumbPosition of scrollbar "Level" > 400 then doOverflow
```

Comments

Use the scrollbar object type to display the current level or the progress of a task, or to let the user set a level or amount.

Comments:

A scrollbar can be displayed as a standard scroll bar, a progress bar, or a sliding control, depending on the setting of the scrollbar's style property. Scrollbars can be vertical or horizontal, depending on whether their height or width is greater.

A scrollbar is contained in a card, group, or background. Scrollbars cannot contain other objects.

The scrollbar object has a number of properties and messages associated with it. To see a list of messages that can be sent to a scrollbar as a result of user actions or internal Revolution events, open the "Transcript Language Dictionary" page of the main Documentation window, and choose "Scrollbar Messages" from the Show menu at the top. To see a list of all the properties a scrollbar can have, choose "Scrollbar Properties" from the Show menu.

scrollbarBeginning

message

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

mouseDown message, scrollbarDrag message, scrollbarEnd message, scrollbarLineDec message, scrollbarPageDec message, shiftKey function, startValue property, thumbPosition property

Summary

Sent to a scrollbar when the user Shift-clicks its decrease arrow.

Syntax

scrollbarBeginning start

Example Code

```
on scrollbarBeginning theStartValue
  -- this scroll bar controls what's in a field
  put theStartValue into field "Count"
end scrollbarBeginning
```

Comments

Handle the scrollbarBeginning message if you want to respond to the use of Shift-click to move the scrollbar thumb directly to the beginning (the top or left) of the scrollbar.

Parameters:

The start is the new position of the scrollbar thumb. This is the same as the scrollbarÆs startValue property.

Comments:

Dragging the scrollbar thumb to the beginning position, or otherwise moving it to the beginning, does not send a scrollbarBeginning message.

If the scrollbarBeginning message is not handled, a scrollbarDrag message is sent in addition.

scrollbarDrag

message

Synonyms

Objects

field, scrollbar, group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

mouseDown message, scrollbarBeginning message, scrollbarEnd message, scrollbarLineDec message, scrollbarLineInc message, scrollbarPageDec message, scrollbarPageInc message, thumbPosition property

Summary

Sent to a field, scrollbar, or group when the user drags the scrollbar thumb or when a text selection causes a field to scroll.

Syntax

scrollbarDrag newPosition

Example Code

```
on scrollbarDrag newValue -- display current line number
  put newValue div the textHeight of me + 1 into field "Line Number"
end scrollbarDrag
```

Comments

Handle the scrollbarDrag message if you want to respond to the user dragging the scrollbar thumb: for example, to update a counter on the card.

Parameters:

The newPosition is the new position of the scrollbar thumb. This parameter is passed only if the object is a scrollbar.

Comments:

If the user clicks one of the arrows or the gray region of the scrollbar, and the appropriate message is not handled anywhere in the message path, a scrollbarDrag message is sent. In other words, you can use a scrollbarDrag handler to handle all scrollbar movements.

If the select command is used to select text within a scrolling field, and this causes the field to scroll to bring the selected text into view, a scrollbarDrag message is sent to the field.

scrollbarEnd

message

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

endValue property, mouseDown message, scrollbarBeginning message, scrollbarDrag message, scrollbarLineInc message, scrollbarPageInc message, shiftKey function, thumbPosition property

Summary

Sent to a scrollbar when the user Shift-clicks its increase arrow.

Syntax

scrollbarEnd end

Example Code

```
on scrollbarEnd -- use a scroll bar to navigate among cards
  visual effect scroll up
  go last card
end scrollbarEnd
```

Comments

Handle the scrollbarEnd message if you want to respond to the use of Shift-click to move the scrollbar thumb directly to the end (the bottom or right) of the scrollbar.

Parameters:

The end is the new position of the scrollbar thumb. This is the same as the scrollbarÆs endValue property.

Comments:

Dragging the scrollbar thumb to the end position, or otherwise moving it to the end, does not send a scrollbarEnd message.

If the scrollbarEnd message is not handled, a scrollbarDrag message is sent in addition.

scrollbarFactor

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

endValue property, hScroll property, startValue property, vScroll property

Summary

Equal to 65535.

Syntax

Example Code

```
get scrollbarFactor
```

Comments

This constant is obsolete and no longer used in Transcript.

scrollbarLineDec

message

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

endValue property, lineIncrement property, mouseDown message, scrollbarBeginning message, scrollbarLineInc message, scrollbarPageDec message, startValue property, thumbPosition property

Summary

Sent to a scrollbar when the user clicks the decrease arrow, scrolling backward one line.

Syntax

scrollbarLineDec newPosition

Example Code

```
on scrollbarLineDec newCharNumber
  -- use a scrollbar to scroll through data, one character at a time
  put char newCharNumber of field "Data" into field "Display"
end scrollbarLineDec
```

Comments

Handle the scrollbarLineDec message if you want to respond to the use of the top or left scrollbar arrow.

Parameters:

The newPosition is the new position of the scrollbar thumb.

Comments:

Dragging the scrollbar thumb does not send a scrollbarLineDec message.

If the scrollbarLineDec message is not handled, a scrollbarDrag message is sent in addition.

scrollbarLineInc

message

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

endValue property, lineIncrement property, mouseDown message, scrollbarEnd message, scrollbarLineDec message, scrollbarPageInc message, startValue property, thumbPosition property

Summary

Sent to a scrollbar when the user clicks the increase arrow, scrolling forward one line.

Syntax

scrollbarLineInc newPosition

Example Code

```
on scrollbarLineInc theCard -- use a scrollbar to navigate through cards
  go card theCard -- corresponding to the new position
end scrollbarLineInc
```

Comments

Handle the scrollbarLineInc message if you want to respond to the use of the bottom or right scrollbar arrow.

Parameters:

The newPosition is the new position of the scrollbar thumb.

Comments:

Dragging the scrollbar thumb does not send a scrollbarLineInc message.

If the scrollbarLineInc message is not handled, a scrollbarDrag message is sent in addition.

scrollbarPageDec

message

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

endValue property, mouseDown message, pageIncrement property, scrollbarBeginning message, scrollbarLineDec message, scrollbarPageInc message, startValue property, thumbPosition property

Summary

Sent to a scrollbar when the user clicks above or left of the scrollbar thumb, scrolling backward one screenful.

Syntax

scrollbarPageDec newPosition

Example Code

```
on scrollbarPageDec
  go previous card
end scrollbarPageDec
```

Comments

Handle the scrollbarPageDec message if you want to respond to the user clicking in the gray region above or to the left of the scrollbar thumb.

Parameters:

The newPosition is the new position of the scrollbar thumb.

Comments:

Dragging the scrollbar thumb does not send a scrollbarPageDec message.

If the scrollbarPageDec message is not handled, a scrollbarDrag message is sent in addition.

scrollbarPageInc

message

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

endValue property, mouseDown message, pageIncrement property, scrollbarEnd message, scrollbarLineInc message, scrollbarPageDec message, startValue property, thumbPosition property

Summary

Sent to a scrollbar when the user clicks below or right of the scrollbar thumb, scrolling forward one screenful.

Syntax

scrollbarPageInc newPosition

Example Code

```
on scrollbarPageInc theMovieNumber
    set the filename of player "Display"

    to line theMovieNumber of field "Movies"
end scrollbarPageInc
```

Comments

Handle the scrollbarPageInc message if you want to respond to the user clicking in the gray region below or to the right of the scrollbar thumb.

Parameters:

The newPosition is the new position of the scrollbar thumb.

Comments:

Dragging the scrollbar thumb does not send a scrollbarPageInc message.

If the scrollbarPageInc message is not handled, a scrollbarDrag message is sent in addition.

scrollbarWidth

property

Synonyms

Objects

field, group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

vScroll property, vScrollbar property, width property

Summary

Specifies the width of the scrollbar associated with a scrolling field or group.

Syntax

set the scrollbarWidth of {field | group} to pixels

Example Code

```
set the scrollbarWidth of last field to 20
```

Comments

Use the scrollbarWidth property to specify how much of a scrolling field's or group's width is taken up by the vertical scrollbar.

Value:

The scrollbarWidth of a field or group is an integer between 1 and the object's width.

By default, the scrollbarWidth property of newly created objects is 16 on Mac OS and OS X systems, and 20 on Unix and Windows systems.

Comments:

Setting a field's or group's vScrollbar or hScrollbar property to true changes its scrollbarWidth to 16 on Mac OS systems, and to 20 on Unix and Windows systems.

If a field's vScrollbar is false and its style is not `scrolling`, its scrollbarWidth property has no effect. If a group's vScrollbar is false, its scrollbarWidth has no effect.

seconds

function

Synonyms

sec, secs

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

convert command, date function, milliseconds function, second keyword, ticks function, time function

Summary

Returns the number of seconds since the start of the eon.

Syntax

the [long] seconds

seconds()

Example Code

```
the seconds -- might return 961694002
```

```
the long seconds -- might return 961694002.834503
```

Comments

Use the seconds function to time events.

Value:

The seconds function returns a positive integer.

If the long seconds form is used, the seconds function returns a positive number with up to six digits after the decimal point.

Comments:

The seconds function returns the total number of seconds since midnight, January 1, 1970 GMT.

The long seconds form returns a value with a precision to a millionth of a second. However, this value is not normally accurate past the third decimal place because of processor delays.

seconds

keyword

Synonyms

second,secs,sec

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

last keyword, middle keyword, milliseconds keyword, number property, seconds function, ticks keyword, two constant

Summary

Designates the second member of a set, the number of seconds in a time period, or a format used with the convert command.

Syntax

Example Code

```
send "mouseUp" to me in 10 seconds
go to second background
get the second item of the abbrev date
convert it to seconds
```

Comments

Use the seconds keyword to designate a time period with the wait or send commands, or to designate the second member of a set in an object reference or chunk expression., or to convert a date or time to the number of seconds since the start of the eon.

Comments:

When used with the wait or send commands, the seconds keyword designates a time period measured in seconds.

The synonym second can also be used to designate the second member of a set, specifying any object whose number property is 2. It can also be used to designate the second chunk in a chunk expression. The word the is optional when using the second keyword in this sense. The synonyms seconds, secs, and sec cannot be used in this way.

secureMode

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

launch command, open file command, open process command, shell function

Summary

Disables the application's ability to access files and run programs.

Syntax

set the secureMode to true

Example Code

```
set the secureMode to true
```

Comments

Use the secureMode property to lock down file access in situations where security is required: for example, for a kiosk application or web server.

Value:

The secureMode is true or false.

By default, the secureMode property is set to false.

Comments:

If the secureMode property is set to true, the application cannot use the get, put, open file, read from file, or write to file commands to gain access to local files. The application cannot run programs with the shell function, the open process command, or the launch command. On Windows systems, it cannot use the deleteRegistry, queryRegistry, or setRegistry functions to access the Windows system registry.

The application can still access remote files with the URL keyword if the secureMode is true. The application can also open (but not save) stack files.

If the application is started from a Unix or Windows command line, this property can be set to true on startup by using the -f option.

Important! Once the secureMode property is set to true, it cannot be set back to false. To change it back to true, you must quit and restart the application.

seek

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close file command, open file command, read from file command, relative keyword, write to file command

Summary

Locates a position in a file, to be used by the read from file and write to file commands.

Syntax

seek {to | rel[ative]} byteNumber in file filePath

Example Code

```
seek to 100 in file "test.txt"  
seek relative -10 in file outputData
```

Comments

Use the seek command to prepare to read from or write to an open file at a specific position in the file.

Parameters:

The byteNumber is an integer representing the position in the file.

The filePath specifies the name and location of the file you want to seek in. It must be the same as the path you used with the open file command. The filePath is case-sensitive, even on platforms where file names are not case-sensitive.

Comments:

The next read from file or write to file command will start reading or writing at the position specified by the seek command.

The seek to form sets the file position to byteNumber bytes from the start of the file. The seek relative form sets the file position to byteNumber bytes from the current position. The position is set by either a previous seek command, or a read from file or write to file command.

select

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

click command, copy command, cut command, delete command, focus command, get command, paste command, put command, selection keyword, selectedObjectChanged message, selectionChanged message, selectionHandleColor property, selectionMode property, How to change the highlight state of a checkbox or button, How to change the selected item in an option menu or combo box, How to select a card, Recipe for shifting the selection to the right or left, Edit menu > Select All, Edit menu > Invert Selection

Summary

Selects part of the text of a field, or places the insertion point in a field, or removes the insertion point, or selects one or more objects.

Syntax

select [before | after] {text | chunk} of field
select empty
select objectList

Example Code

```
select text of field ID 3
select after word 34 of field myField
select empty
select button "OK"
select scrollbar 1 and image "Scroll" and field "Label"
```

Comments

Use the select command to select objects or text in order to change them, or to deselect all objects and text.

Parameters:

The chunk is a chunk expression specifying a portion of the field.

The field is a field reference.

The objectList consists of one or more object references, separated by the word and.

Comments:

If you select text in a field, the field scrolls (if necessary) to show the selected chunk.

Use the select before or select after form to position the insertion point at a particular location in a field.

Use the select empty form to deselect all objects and text. To select an object without deselecting any objects that are already selected, use the selected property instead of the select command.

To select lines in a list field, set the field's `hilitedLine` property instead of using the select command.

To select text in a field, the field's `traversalOn` property must be set to true.

select
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

choose command, selectedImage function, tool function

Summary

Designates the paint tool used to select a rectangular area of an image.

Syntax

Example Code

```
choose select tool
```

Comments

Use the select keyword to select an area of an image.

Comments:

When using the Select tool, the cursor is a crosshairs shape. This tool is used to select a rectangle in an image, by clicking one corner and dragging to the opposite corner.

You can drag a selected rectangle to another location in the image or cut, copy, or delete it.

selected

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

choose command, focus command, select command, selectedImage function, selectedObject function, selectedObjectChanged message

Summary

Specifies whether an object is selected.

Syntax

set the selected of object to {true | false}

Example Code

```
if the selected of button "Group" then select button "Ahhh"
```

Comments

Use the selected property to select an object or find out whether the user has selected it with the Pointer tool.

Value:

The selected of an object is true or false.

Comments:

You can select an object by setting its selected property to true, and deselect it by setting its selected to false.

selectedButton

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

family property, hilitedButton property, radioBehavior property

Summary

Returns the name of the currently highlighted button in a family.

Syntax

the selectedButton of [card | background] family familyNumber

Example Code

```
the selectedButton of family 3  
put the selectedButton of family myFamily into thisButton
```

Comments

Use the selectedButton function to find out the current setting of a radio-button cluster controlled by the family property.

Parameters:

The familyNumber is a positive integer.

Value:

The selectedButton function returns a string of the form domain button buttonNumber, where domain is either card or background.

Comments:

This function exists for compatibility with imported HyperCard stacks. You can use the radioBehavior property to cause a group consisting of buttons to behave as a radio cluster, and use the hilitedButton property to find out which button in the group is highlighted.

selectedChunk

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickChunk function, foundChunk function, mouseChunk function, select command, selectedField function, selectedLine function, selectedLoc function, selectedText function, Why is the selected text deselected?, Why is the selection lost when clicking a button?

Summary

Returns a chunk expression describing the location of the text selection or insertion point.

Syntax

the selectedChunk

selectedChunk

Example Code

```
the selectedChunk  
put the selectedChunk into storedChunk
```

Comments

Use the selectedChunk function to determine which text is selected.

Value:

The selectedChunk function returns a chunk expression of the form
char startChar to endChar of field fieldNumber

Comments:

The return value reports the selected text: the startChar is the first character of the selection, and the endChar is the last character.

If no text is selected but the text insertion point is in a field, the startChar is the character after the insertion point, and the endChar is the character before the insertion point. In this case, the endChar is one less than the startChar.

If there is no insertion point or text selection, the selectedChunk function returns empty.

To get actual text that is selected, use the `selectedText` function.

selectedColor

property

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

Summary

The selectedColor property is reserved for internal use.

Syntax

Example Code

Comments

selectedField

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickField function, foundField function, selectedChunk function, selectedLine function, selection keyword, Why is the selected text deselected?, Why is the selection lost when clicking a button?

Summary

Returns the number of the field with the text selection or insertion point.

Syntax

the selectedField

selectedField()

Example Code

```
the selectedField
```

```
hide the selectedField
```

Comments

Use the selectedField function to determine which field the selection is in.

Value:

The selectedField function returns the number of the field with the selection.

Comments:

If there is no insertion point or text selection, the selectedField function returns empty. Otherwise, the selectedField is the same object as the one specified by the focusedObject function.

selectedImage

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

choose command, select keyword, selected property, selectedField function, selectedObject

Summary

Returns the number of the image which is partly or completely selected with the Select tool.

Syntax

the selectedImage

selectedImage()

Example Code

```
the selectedImage
```

```
put the short name of the selectedImage into field "Current Image"
```

Comments

Use the selectedImage function to find out which image the user is painting in.

Value:

The selectedImage function returns the word "image" followed by an image's number.

If nothing is selected with the select tool, the selectedImage function returns empty.

Comments:

You use the Pointer tool to select entire objects (including images), but you use the Select tool (which is one of the paint tools) to select a rectangular portion of the picture that's contained in an image. The selectedImage function reports on the Select tool's selection, not the Pointer tool's.

To find out which objects are selected with the Pointer tool, use the selectedObject function instead.

selectedLine

function

Synonyms

selectedLines

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickLine function, foundLine function, hilitedLine property, mouseLine function, select command, selectedChunk function, selectedField function, selectedText function, Why is the selected text deselected?, Why is the selection lost when clicking a button?, Recipe for shifting the selection to the right or left

Summary

Returns a chunk expression describing the line or lines in the text selection.

Syntax

the selectedLine

selectedLine()

Example Code

```
the selectedLine
select the selectedLine
```

Comments

Use the selectedLine function to determine which line the selection or text insertion point is in.

Value:

The selectedLine function returns a chunk expression of the form

line lineNumber of field fieldNumber

if the text selection is on one line, or

line startLine to endLine of field fieldNumber

if the text selection crosses multiple lines.

Comments:

A line is defined as a string that's delimited by return characters. A line in a narrow field may wrap around several times, and look like more than one line on screen, but it is still considered a single line.

If there is no insertion point or text selection, the selectedLine function returns empty.

To get a chunk expression describing the location of the selection, use the `selectedChunk` function. To get the selected lines in a list field that allows non-contiguous selections, use the `hilitedLine` property.

selectedLoc

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickLoc function, foundLoc function, mouseLoc function, select command, selectedField function, selectedLine function, selectedText function, selection keyword, topLeft property

Summary

Returns the location of the top left corner of the text selection.

Syntax

the selectedLoc

selectedLoc()

Example Code

```
the selectedLoc
if the selectedLoc is not within the rect of me then unScrollField
```

Comments

Use the selectedLoc function to determine the location in the stack window where the text selection appears.

Value:

The selectedLoc function returns two positive integers separated by a comma.

Comments:

When text is selected, a box around it is filled with the hiliteColor or hilitePattern. The first item of the return value is the horizontal distance in pixels from the left edge of the stack to the left edge of this box. The second item of the return value is the vertical distance from the top edge of the stack to the top of the box.

If the text selection is in a scrolling field and the selection is not in the visible portion of the field, the selectedLoc function is adjusted by the amount of scroll.

selectedObject

function

Synonyms

selectedObjects, selObj, selObjs, selectedGraphic

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

group command, select command, selectedField function, selectedImage function, selectedObjectChanged message, selectedText function, selectionHandleColor property, selectionMode property

Summary

Returns the name of the selected object or objects.

Syntax

the selectedObject
selectedObject()

Example Code

```
the selectedObject  
set the lockText of the selectedObject to true  
repeat with x = 1 to the number of lines of the selectedObjects
```

Comments

Use the selectedObject function to perform an action on all the selected objects.

Value:

The selectedObject function returns a list with the long ID property of each selected object, one per line.

selectedObjectChanged

message

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantSelect property, choose command, mouseDown message, newTool message, select command, selectionChanged message

Summary

Sent to an object when it is selected.

Syntax

selectedObjectChanged

Example Code

```
on selectedObjectChanged -- update a property-display stack
  repeat for each line thisObjectID in the selectedObjects
    send "addToPalette" && thisObjectID to stack "Object Palette"
  end repeat
end selectedObjectChanged
```

Comments

Handle the selectedObjectChanged message if you want to perform some action when an object is selected.

Comments:

When an object is selected, the selectedObjectChanged message is sent to the newly selected object.

When an object is deselected and no other object is selected—for example, when the user clicks the card or chooses the Browse tool—the selectedObjectChanged message is sent to the object being deselected.

selectedText

function

Synonyms

hilitedText

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickText function, foundText function, hilitedLine property, menuHistory property, select command, selectedChunk function, selectedField function, selectedLine function, How to determine the current selection in a menu, Why is the selected text deselected?, Why is the selection lost when clicking a button?, Recipe for shifting the selection to the right or left

Summary

Returns the contents of the text selection.

Syntax

the selectedText [of field | button]

selectedText([field | button])

Example Code

```
the selectedText of field "Chapter Titles"  
put the selectedText into thisSelection
```

Comments

Use the selectedText function to find out what text is selected.

Parameters:

The field is any valid field reference.

The button is any valid button reference.

Value:

The selectedText function returns a string.

Comments:

If a field is specified, the selectedText function returns the selected text in that field. If the fieldÆs listBehavior property is set to true, the selectedText includes any selected lines in the field.

If a button is specified, the `selectedText` function returns the currently selected menu item associated with the button. If the button does not have a menu associated with it, the `selectedText` function returns empty.

If there is no insertion point or text selection and no field or button is specified, the `selectedText` function returns empty.

To get the location of the selected text, use the `selectedChunk` function.

To change the contents of the selection, use the `selection` keyword.

selectGroupedControls

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

select command, selectedObject function, selection keyword, selectionMode property, Edit menu > Select Grouped Controls

Summary

Specifies whether clicking an object in a group selects the object or the entire group.

Syntax

set the selectGroupedControls to {true | false}

Example Code

```
set the selectGroupedControls to (not the selectGroupedControls)
```

Comments

Use the selectGroupedControls property to control what happens when you click a grouped control.

Comments:

If the selectGroupedControls property is set to false, clicking an object in a group with the Pointer tool selects the entire group. You must set the editBackground property to true in order to select, move, resize, and change individual objects in the group.

If the selectGroupedControls is true, clicking an object in a group selects only that object, even if the editBackground is false.

You can change the selectGroupedControls property by choosing Edit menu Select Grouped Controls.

selection

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

mouseColor function, select command, selectedChunk function, selectedImage function, selectedObject function, selectedText function, selectionChanged message, How to determine the current selection in a menu, Why is the selected text deselected?, Why is the selection lost when clicking a button?, Edit menu > Select All, Edit menu > Deselect All, Edit menu > Invert Selection

Summary

The selection is a reference to the currently selected text.

Syntax

the selection

selection()

Example Code

```
put empty into the selection -- deletes the selection
```

Comments

Use the selection keyword to get the currently selected text or to replace the selection.

Comments:

Certain commands—in particular, the cut and copy commands—operate on the selected text.

Note: The selection keyword is implemented internally as a function and appears in the functionNames. However, unlike most functions, the selection lets you put things into it, and behaves like a container rather than a function.

selectionChanged

message

Synonyms

Objects

field, player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentTimeChanged message, endTime property, select command, selectedObjectChanged message, selection keyword, startTime property, Why is the selection lost when clicking a button?

Summary

Sent to a field or player when the selection is changed.

Syntax

selectionChanged

Example Code

```
on selectionChanged -- display selected char number
  put word 2 of the selectedChunk into field "Character Number"
end selectionChanged
```

Comments

Handle the selectionChanged message if you want to perform updates or do some action when the user changes the selected text of a field or the selected portion of a player.

Comments:

The selectionChanged message is sent to a field whenever the user makes a text selection or moves the insertion point in the field, after the selection is changed. (The selectionChanged message is not sent if the user uses the arrow keys to change the selection. To respond to use of the arrow keys, handle the rawKeyUp message and check the selection within the handler.)

The selectionChanged message is sent to a player whenever the user selects a portion of the sound or movie, or when a handler sets the player's startTime or endTime property.

selectionHandleColor

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

borderColor property, colorNames function, penColor property, select command, selectedObject function, About colors and color references, Color Names Reference, Recipe for translating a color name to an RGB numeric triplet

Summary

The selectionHandlerColor property specifies the color of the handle boxes used to resize a selected object.

Syntax

set the selectionHandleColor to {colorName | RGBColor}

Example Code

```
set the selectionHandleColor to "red"
set the selectionHandleColor to "255,0,0"
set the selectionColor to the hiliteColor
```

Comments

Use the selectionHandleColor property to change the appearance of selected objects.

Value:

The selectionHandleColor is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the selectionHandleColor is  0,0,0  (black).

selectionMode

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

drag command, intersect function, select command, selectedObject function, Edit menu > Intersected Selections

Summary

Determines which objects are selected when you drag a rectangle with the Pointer tool.

Syntax

set the selectionMode to {intersect | surround}

Example Code

```
set the selectionMode to intersect
```

Comments

Use the selectionMode property to change the behavior of drag selections.

Value:

The selectionMode is either intersect or surround.

Comments:

When you choose the Pointer tool, click in the window, and drag diagonally to select a rectangle, objects within the rectangle are selected. The selectionMode property determines which objects are included inside the dragged rectangle.

If the selectionMode is set to `intersect`, all objects that are partially or completely inside the rectangle are selected. If the selectionMode is set to `surround`, only objects that are completely inside the rectangle are selected.

Tip: You can also set the selectionMode by choosing Edit menu Intersected Selections.

send

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

call command, cancel command, do command, dynamicPaths property, idle message, pass control structure, pendingMessages function, target function, value function, About messages and the message path, How to call a custom function thatÆs not in the message path, How to schedule a future message, How to trigger a handler, Recipe for 99 Bottles of Beer on the Wall (using send), Recipe for a card slideshow, Recipe for a ôroll creditsö effect

Summary

Sends a message to an object.

Syntax

send message [to object] [in time [seconds | ticks | milliseconds]]

Example Code

```
send "mouseDown" to button "OK"  
send "beep" to me in 20 seconds  
send "goThere" && field "Where To" to stack "Destination" in 30
```

Comments

Use the send command to override the normal message path, or to delay a command until a specified time.

Parameters:

The message is an expression that evaluates to a message name, possibly including parameters.

The object is any object reference. If you donÆt specify an object, the message is set to the object whose handler contains the send command. If you specify a time, you must also specify an object.

The time is an integer. If you donÆt specify a unit of time, the message is sent after time ticks.

Comments:

The message can be either the name of a handler or a Transcript statement. If the message is a literal string containing more than one word (for example, if it includes parameters, or if it is a multi-word command), it must be enclosed in double quotes.

Any parameters are evaluated before they are passed to the send command. For example, the following statement sends the mouseUp message with 2 as the first parameter:

```
send "mouseUp 1+1" to button "Example"
```

All object references in the message are evaluated relative to the target object, not to the object whose script issued the send command. However, if the handler refers to an object that does not exist in the current context, the reference will cause a script error. For example, if the handler contains the statement put the date into field 1, and when the message is sent the user is on a card that does not include any fields, this statement will cause an error.

Important! If an execution error occurs in a handler that's called using the send in time form of the send command, no errorDialog message is sent. This means that it's especially critical to thoroughly debug any handlers that will be called using this method.

If you specify a time, the message is sent to the object after the specified time has elapsed. The message is added to the pendingMessages function. The ID of the message is returned by the result function. To cancel the message, use the cancel command with this ID.

Important! Specifying a time can affect the order in which statements are executed. If you don't specify a time, the message is sent immediately, and any handler it triggers is completed before the rest of the current handler is executed. If you use the send in time form of the send command—even if you specify a time of zero seconds—the current handler finishes executing before the message is sent.

Note: Using the send command is slower than directly executing the commands using the normal message path. For best efficiency, use the send command only when you want to delay the message or when the handler you want to execute is not in the message path.

send to program command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

address property, appleEvent message, reply command, request appleEvent command, request command

Summary

Sends an Apple event to a program.

Syntax

send message to {program | application} programAddress

[with classID] [{with | without} reply]

Example Code

```
send "get field 3" to program "Corporate Zone:Other Mac:Revolution"  
send thisMessage to application "FileMaker" without reply  
send field ID 9 to program someProgram with "GURLGURL"
```

Comments

Use the send to program command to cause another application to take some action via the dosc Apple event or another Apple event you specify.

Parameters:

The message is the message you want the other application to execute (the data attached to the Apple event you are sending). Its exact format and meaning depends on the other application.

The programAddress is the AppleTalk address of the other program. The AppleTalk address consists of three parts, separated by colons: the zone the other computer is in, the name of the computer, and the name of the target program. If the other computer is in the same zone as yours, you can omit the zone. If the other program is running on the same computer, you can omit both the zone and the computer name.

The classID is a string consisting of the 4-character Apple event class and the 4-character event ID. If you don't specify a classID, Revolution uses "miscdosc": the misc class and the dosc event.

Comments:

If the program supports the Apple event, it performs the command. If the program sends back a response, it is placed in the result function.

The to application and to program forms are synonyms.

If you specify the with reply form, the handler pauses until the application sends back a response. If you specify the without reply form, the handler continues immediately without waiting. If you don't specify either form, the with reply form is used.

Tip: To display the process browser dialog box and allow the user to choose a running program, use the following statement:

```
do "choose application" as AppleScript
```

For more information about Apple events, see Apple Computer's technical documentation, Inside Macintosh: Interapplication Communication, located at
<<http://developer.apple.com/techpubs/mac/IAC/IAC-2.html>>.

serialControlString

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close file command, COM1: keyword, driverNames function, modem: keyword, open driver command, open file command, printer: keyword

Summary

The serialControlString specifies the settings of a serial port.

Syntax

set the serialControlString to settingsList

Example Code

set the serialControlString to storedSerial

Comments

Use the serialControlString property to set a serial port before use.

Value:

The serialControlString consists of one or more settings, separated by spaces. Each setting consists of the setting name, an equals sign (=), and the value.

The possible settings are as follows:

- ò BAUD=number: the port's baud rate
- ò PARITY=N, O, or E: no parity, odd parity, or even parity
- ò DATA=numberOfDataBits
- ò STOP=numberOfStopBits
- ò to=on or off: use timeouts
- ò xon=on or off: software handshaking
- ò odsr=on or off: (output) data set ready
- ò octs=on or off: (output) clear to send
- ò dtr=on or off: data terminal ready
- ò rts=on or off: ready to sent
- ò isdr=on or off: (input) data set ready

By default, the serialControlString is set to BAUD=9600 PARITY=N DATA=8 STOP=1.

Comments:

To set a serial port's settings, first set the serialControlString to the desired settings. Then open the serial port using the open file command or the open driver command.

On Mac OS systems, the serialControlString property can be used to set the printer or modem ports. On Windows systems, the serialControlString can be used to set the COM ports, up to COM9. On OS X systems, the serialControlString can be used to set a serial device returned by the driverNames function.

The format of the serialControlString is compatible with the extended MS-DOS `mode` command.

Changes to Transcript:

Support for OS X and Unix serial devices was added in version 2.0.

set
command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

customKeys property, customPropertySet property, get command, keys function, propertyNames function, setProp control structure

Summary

Assigns a value to a property.

Syntax

set [the] property [of object] to value

Example Code

```
set the textFont of button "OK" to "Arial"  
set the cursor to watch
```

Comments

Use the set command to change the setting of a property or custom property.

Parameters:

The property is a built-in property name or custom property name.

The object is any object reference. If you don't specify an object, the property must be a global property.

The value is any expression that evaluates to a string.

Comments:

If you specify a property that doesn't exist, Revolution creates a custom property with that name for the object.

set
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems
Not supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of last button to set
```

Comments

Use the set keyword to turn an object white.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

When the set mode is used, each component of the color underneath the object's red, green, and blue is raised to be equal to white.

The set mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear as though their ink were set to reverse.

setProp

control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

customProperties property, getProp control structure, on control structure, pass control structure, properties property, propertyName function, set command, About custom properties and custom property sets, About the structure of a script, Recipe for setting the red channel of an object, Recipe for switching between styled text and HTML source

Summary

Handles the trigger sent to an object when you change one of its custom properties.

Syntax

```
setProp propertyName newValue  
    statementList  
end propertyName
```

Example Code

Comments

Use the setProp control structure to check the range of a custom property that is being set, or to change other properties or do other tasks at the same time a custom property is set.

Form:

The first line of a setProp handler consists of the word `setProp` followed by the name of the custom property.

The last line of a setProp handler consists of the word `end` followed by the property's name.

Parameters:

The propertyName is a string up to 65,535 characters in length.

The newValue is a string.

The statementList consists of one or more Transcript statements, and can also include if, switch, try, or repeat control structures.

Comments:

A setProp handler can contain any set of Transcript statements.

The propertyName is the name of the custom property whose value is being changed with the set command.

Note: You cannot use a setProp handler to intercept a built-in property. The setProp control structure can be used only for custom properties.

The setProp trigger passes through the message path, the same as any other message, so a setProp handler for an object can be located in the object's script or in the script of any object further in the message path. For example, a setProp handler for a card property may be located in the script of the stack that the card belongs to.

If you use the set command within a setProp control structure to set the same custom property for the current object, no setProp trigger is sent to the object. (This is to avoid runaway recursion, where the setProp handler triggers itself.)

This is only the case for the custom property that the current setProp handler applies to. Setting a different custom property sends a setProp trigger. So does setting the handler's custom property for an object other than the one whose script contains the setProp handler.

Caution! If a setProp handler in one object's script sets the custom property for a different object, and the first object is in the second object's message path, a runaway recursion will result. For example, if the following handler is in a card script, and you set the myCustomProperty of a button on the card, runaway recursion will result:

```
setProp myCustomProperty newValue
  set the myCustomProperty of the target to newValue + 1
  -- Because the target is the button, and this handler is in
  -- the card, the above statement sends another setProp trigger
  -- to the button.
end myCustomProperty
```

To avoid this problem, set the lockMessages property to true before setting the custom property.

You can include as many setProp handlers in a script as you need. The property that a setProp handler controls is determined by the propertyName parameter in the first line of the handler. (If a script contains two setProp handlers for the same property, the first one is used.)

If the custom property you want to control is in a custom property set, use array notation in the first line of the setProp handler, as in the following example:

```
setProp mySet[thisProperty] newValue
  if thisProperty is "that" then put newValue into me
end setProp
```

The above setProp handler responds to changes in the custom property named `that`, which is a member of the custom property set named `mySet`.

Important! You must either include the pass control structure or set the property explicitly in a setProp handler, if you want Revolution to set the custom property. Otherwise, the setProp handler traps the call to set the property, and its value is not changed.

Changes to Transcript:

In version 2.0 and later, setting a custom property (other than the one that the current setProp handler applies to) inside a setProp handler sends a setProp trigger. In previous versions, setting a custom property within a setProp handler never sent a setProp trigger, regardless of which custom property you set.

setRegistry

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

deleteRegistry function, platform function, queryRegistry function

Summary

Sets an entry in the Windows system registry.

Syntax

setRegistry(keyPath,value[,type])

Example Code

```
setRegistry("HKEY_CLASSES_ROOT\txt", "NotePad")
if setRegistry(myEntry & "/",myValue) then open file myFile
setRegistry(theKey,empty)
```

Comments

Use the setRegistry function to change the behavior of Windows.

Parameters:

The keyPath parameter is the path to a registry entry.

The value is the new setting for the entry specified by the keyPath.

The type is one of the following: `binary`, `dword`, `dwordlittleendian`, `dwordbigendian`, `expandable`, `link`, `multisize`, `none`, `resource`, `string`, or `size`. If you don't specify a type, `string` is used.

Value:

The setRegistry function returns true if the setting was successfully changed, false otherwise.

Comments:

The first part of the keyPath should be one of the predefined handle values. If the keyPath ends in `0`, the new setting becomes the default value. For example, if you want files ending in `0.rev` to automatically launch Revolution, use the following command:

```
get setRegistry("HKEY_CLASSES_ROOT\Revolution")
```

If the key doesn't exist, the setRegistry function creates it in the registry.

To delete a subkey, set the subkey's value to empty.

If the type is a binary type, make sure the value is binary. (You can encode a string as binary data using the binaryEncode function.)

If Windows sends an error message to the application, the result function returns the error message.

On Mac OS and Unix systems, the setRegistry function returns "not supported".

Caution! Be careful to use only carefully debugged entries with the setRegistry function. Changing entries in the Windows registry to invalid values can cause the system to behave unexpectedly or stop functioning altogether.

Changes to Transcript:

The type parameter was added in version 2.0. In previous versions, the type information could not be set.

setResource

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

copyResource function, deleteResource function, getResource function, getResources function, resfile keyword

Summary

Places data in a specified resource in a Mac OS file.

Syntax

setResource(destinationFile,resourceType,[resID],[resName],flagsList,data)

Example Code

```
setResource("Include","BNDL",129,"document",RL,the bundleData of me)
setResource(it,"ICON",128,,getResource(it,"ICON",130))
```

Comments

Use the setResource function to create a resource or change its data.

Parameters:

The destinationFile is the location and name of the file that contains the resource you want to set. If you specify a name but not a location, Revolution assumes the file is in the defaultFolder.

The resourceType is the 4-character type of the resources you want to change.

The resID is an integer or an expression that evaluates to an integer.

The resName is a string or an expression that evaluates to a string.

The flagsList is a list that can contain one or more flag characters. The possible resource flags are as follows:

S	System heap
U	Purgeable
L	Locked
P	Protected

R Preload
C Compressed resource

The flags may be listed in any order. If a character is included, its corresponding resource flag is set to true. If the character is not included in the flagsList, its corresponding resource flag is set to false. If the flagsList is empty, all the flags are set to false.

The data is text or binary data, formatted appropriately for the resource type.

Value:

The setResource function returns empty.

Comments:

If the destinationFile does not exist, the setResource function creates the file. If the destinationFile exists but has no resource fork, the setResource function creates the resource fork and copies the resource to it.

If the destinationFile is open, the result is set to "Can't open resource fork".

If the specified resource already exists, the setResource function replaces the data in the resource with the data. Otherwise, the setResource function creates the resource.

You must specify either a resID or resName or both. If you specify one but not the other, the setResource function looks for an existing resource with the specified name or ID, and replaces its contents without changing the existing ID or name.

The setResource function is most useful when used with the getResource function. Use the getResource function to obtain resource data, process it as desired, then use the setResource function to change the resource.

Caution! Resource data must be in the specific format appropriate to the resource type. If you set a resource to data that is incompatible with its resource type, you may damage the resource or the entire resource fork of the file. For information on the format of standard resource types, see Apple Computer's technical documentation, Inside Macintosh, located at <http://developer.apple.com/techpubs/macos8/mac8.html>.

seven
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, cross constant, seventh keyword

Summary

Equivalent to the number 7.

Syntax

Example Code

```
subtract seven from dayOfWeek
```

Comments

Use the seven constant when it is easier to read than the numeral 7.

seventh

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

first keyword, last keyword, middle keyword, number property, seven constant

Summary

Designates the seventh member of a set.

Syntax

Example Code

```
set the textFont of seventh button to nextFont  
put "Hello!" into seventh line of field "Greetings"
```

Comments

Use the seventh keyword in an object reference or chunk expression.

Comments:

The seventh keyword can be used to specify any object whose number property is 7. It can also be used to designate the seventh chunk in a chunk expression.

The word the is optional when using the seventh keyword.

shadow

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

style property, transparent keyword, Object menu > New Control > Shadow Button

Summary

Used with the style property to indicate a field or button with a drop shadow.

Syntax

Example Code

```
set the style of field "Label" to shadow
```

Comments

Use the shadow keyword to create a field or button with a drop shadow.

Comments:

Setting a control's style property to `shadow` sets its shadow property to true.

shadow

property

Synonyms

Objects

control, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

decorations property, shadowColor property, shadowOffset property, shadowPattern property, showBorder property, threeD property

Summary

Specifies whether a button, field, or stack window is drawn with a drop shadow.

Syntax

set the shadow of object to {true | false}

Example Code

```
set the shadow of field ID 18 to true
set the shadow of this stack to false
```

Comments

Use the shadow property to draw a drop shadow.

Value:

The shadow of an object is true or false.

By default, the shadow of a newly created stack is set to true. The shadow of a newly created control is false by default, but may be true if it is created by the development environment. (For example, choosing Object menu>New Control>Shadow Button creates a button whose shadow is true.)

Comments:

While the shadow property can be set for any control, it affects the appearance of only fields and buttons. Other controls do not display a drop shadow, regardless of the setting of their shadow property.

The shadow of a stack window is drawn by the operating system. On Mac OS, Unix, and Windows systems, the setting of a stack's shadow property has no effect.

The setting of this property for a stack affects the stack's decorations property, and vice versa. Setting a stack's shadow property determines whether its decorations property includes "noShadow" (or is

ôdefaultö, for window styles that normally include a drop shadow). Conversely, setting a stackÆs decorations property sets its shadow to true or false depending on whether the decorations includes ônoShadowö (or is ôdefaultö).

By default, the shadow extends 4 pixels to the left and below the object. You can change the size and direction of the drop shadow with the shadowOffset property. (The shadowOffset has no effect on stack windows.)

Changes to Transcript:

The ability to specify the shadow of a stack was added in version 2.1.

shadowColor

property

Synonyms

seventhColor

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundColor property, borderColor property, bottomColor property, colorNames function, colors property, effective keyword, focusColor property, foregroundColor property, hiliteColor property, owner property, shadow property, shadowPattern property, topColor property, About colors and color references, Color Names Reference, Why don't buttons respect my color settings?, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies the color of an object's drop shadow or the background of a scrollbar.

Syntax

set the shadowColor of object to {empty | colorName | RGBColor}

Example Code

```
set the shadowColor of last scrollbar to myBgColor
set the shadowColor of button "OK" to "gray"
set the shadowColor of the mouseControl to "#336699"
```

Comments

Use the shadowColor property to specify the drop shadow color of a field or button or the background color of a scrollbar.

Value:

The shadowColor of an object is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the shadowColor for all objects is empty.

Comments:

Setting the shadowColor of an object to empty allows the shadowColor of the object's owner to show through. Use the effective keyword to find out what color is used for the object, even if its own shadowColor is empty.

The setting of the shadowColor property has different effects, depending on the object type:

ò The shadowColor of a stack, card, or group determines the shadowColor of each object in the stack, card, or group that does not have its own shadowColor.

ò The shadowColor of a button is used for the button's drop shadow. If the button's style is menu (unless the menuMode is tabbed), the shadowColor has no effect. If the button is a tabbed button, the shadowColor is always used for the inactive tabs; otherwise, if the button's shadow property is false, the shadowColor has no effect.

ò The shadowColor of a field determines the color of the field's drop shadow. If the field's shadow property is false, the shadowColor has no effect.

ò The shadowColor of a scrollbar determines the background color of the scrollbar.

ò The shadowColor of a graphic, player, audio clip, video clip, or EPS object has no effect.

ò The shadowColor of an image is the seventh color in the image's color palette.

If an object's shadowPattern is set, the pattern is shown instead of the color specified by the shadowColor.

shadowOffset

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

borderWidth property, shadow property, shadowColor property, shadowPattern property, threeD property

Summary

Specifies the size and direction of an object's drop shadow.

Syntax

set the shadowOffset of object to pixels

Example Code

```
set the shadowOffset of button "Prospects" to 2
set the shadowOffset of field "Help" to -6
```

Comments

Use the shadowOffset property to change the appearance of drop shadows.

Value:

The shadowOffset of an object is an integer between -128 and 127.

By default, the shadowOffset of a newly created object is 4.

Comments:

Because this also changes the position of the imaginary light source that casts the shadow, the shadowOffset of all objects on a card should usually be the same.

The shadowOffset specifies how far the shadow extends from the edge of the object. If the shadowOffset is positive, the imaginary light source is at the upper left corner of the screen, so the drop shadow falls below and to the right of the object. If the shadowOffset is negative, the direction of the light source is reversed, and the drop shadow falls above and to the left of the object.

If the object's shadow property is false, the shadowOffset has no effect.

shadowPattern

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backgroundPattern property, borderPattern property, bottomPattern property, brushPattern property, effective keyword, focusPattern property, foregroundPattern property, hilitePattern property, patterns property, shadowColor property, topPattern property

Summary

Specifies the pattern of an object's drop shadow.

Syntax

set the shadowPattern of object to {patternNumber | imageID | empty}

Example Code

```
set the shadowPattern of next card to empty
set the shadowPattern of last button to 409
```

Comments

Use the shadowPattern property to specify the drop shadow pattern of a field or button or the background pattern of a scrollbar.

Value:

The shadowPattern of an object is a pattern specifier.

A patternNumber is a built-in pattern number between 1 and 164. (These patterns correspond to Revolution's built-in patterns 136 to 300.)

An imageID is the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the shadowPattern for all objects is empty.

Comments:

Pattern images can be color or black-and-white.

Cross-platform note: To be used as a pattern on Mac OS systems, an image must be 128x128 pixels or less, and both its height and width must be a power of 2. To be used on Windows and Unix systems, height and width must be divisible by 8. To be used as a fully cross-platform pattern, both an image's dimensions should be one of 8, 16, 32, 64, or 128.

The shadowPattern of controls is drawn starting at the control's upper right corner: if the control is moved, the pattern does not shift.

Setting the shadowPattern of an object to empty allows the shadowPattern of the object's owner to show through. Use the effective keyword to find out what color is used for the object, even if its own shadowPattern is empty.

The setting of the shadowPattern property has different effects, depending on the object type:

- ò The shadowPattern of a stack, card, or group determines the shadowPattern of each object in the stack, card, or group that does not have its own shadowPattern.

- ò The shadowPattern of a button is used for the button's drop shadow. If the button's style is menu (unless the menuMode is tabbed), the shadowPattern has no effect. If the button is a tabbed button, the shadowPattern is always used for the inactive tabs; otherwise, if the button's shadow property is false, the shadowPattern has no effect.

- ò The shadowPattern of a field determines the pattern used for the field's drop shadow. If the field's shadow property is false, the shadowPattern has no effect.

- ò The shadowPattern of a scrollbar determines the background pattern of the scrollbar.

- ò The shadowPattern of a graphic, image, player, audio clip, video clip, or EPS object has no effect.

If an object's shadowPattern is set, the pattern is shown instead of the color specified by the shadowColor.

shadowPixel

property

Synonyms

seventhPixel

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

backgroundPixel property, borderPixel property, bottomPixel property, colorMap property, focusPixel property, foregroundPixel property, hilitePixel property, screenColors function, shadowColor property, topPixel property, Recipe for translating a color name to an RGB numeric triplet

Summary

Specifies which entry in the color table is used for the color of an object's drop shadow or the background of a scrollbar.

Syntax

set the shadowPixel of object to colorNumber

Example Code

```
set the shadowPixel of this card to the short number of this card
```

Comments

Use the shadowPixel property to change the shadow color of an object when the bit depth of the screen is 8 bits (256 colors) or less.

Value:

The shadowPixel of an object is an integer between zero and (the screenColors - 1). It designates an entry in the current color table.

By default, the shadowPixel for all objects is empty.

Comments:

The shadowPixel property specifies which entry in the color table is used for an object's shadow color. It is similar to the shadowColor property, but is specified as an entry in the current color table, rather than as a color reference.

The color table can be set by changing the colorMap property.

sharedHilite

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hilite property, hilitedButton property, radioButton property, sharedText property, How to give a checkbox a different state on each card

Summary

Specifies whether a grouped buttonÆs hilite property is the same on all cards with that group.

Syntax

set the sharedHilite of button to {true | false}

Example Code

```
set the sharedHilite of button "Option 1" to false
```

Comments

Use the sharedHilite property to cause a button (such as a radio button or checkbox) to display the same state wherever it is shown.

Value:

The sharedHilite of a button is true or false.

By default, the sharedHilite of newly created buttons is set to true.

Comments:

If a buttonÆs sharedHilite property is set to true, the hilite property of that button is the same on all cards where the button appears. Highlighting the button on one card highlights it on all of them. If the buttonÆs sharedHilite is false, it can have a different hilite property on each card where it appears, and highlighting a button highlights it only on the current card.

Set the sharedHilite property to false for radio buttons and checkboxes (and for other buttons whose hilite is retained to display a setting or a state) if the state the button displays is different for each card.

sharedText

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

dontSearch property, sharedHilite property, How to display the same text on multiple cards

Summary

Specifies whether a grouped field's text is the same on all cards with that group.

Syntax

set the sharedText of field to {true | false}

Example Code

```
set the sharedText of field "Section Marker" to true
```

Comments

Use the sharedText property to create labels or other fields with static text (the same on all cards where the field appears).

Value:

The sharedText of a field is true or false.

By default, the sharedText property of newly created fields is set to false.

Comments:

If a field's sharedText property is set to true, the content of that field is the same on all cards where the field appears. Changing the text on one card changes it on all the cards.

If the field's sharedText is false, it can have different text on each card where it appears, and changing the field's text changes it only on the current card.

If the field is a card field (not part of a group), its sharedText property has no effect.

sheet

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

answer command, ask command, dialogData property, go command, modal command, mode property, modeless command, palette command, topLevel command, About windows, palettes, and dialog boxes, How to create a custom dialog box

Summary

Displays a stack as a sheet dialog box.

Syntax

sheet stack [in parentStack]

Example Code

```
sheet stack "Document Settings"
sheet stack (the nextDialog of me)
sheet me in stack "Overview"
```

Comments

Use the sheet command to display a stack as a custom sheet.

Parameters:

The stack is any stack reference.

The parentStack is any open stack that is not being displayed as a sheet.

Comments:

The sheet command opens the stack as a sheet in the window of the specified parentStack. If you don't specify a parentStack, the sheet is displayed in the window of the defaultStack.

The stack's rectangle and location properties are ignored.

While a sheet dialog box is open, nothing else can be done in the stack the sheet is displayed in. Because of this, you should use sheets only when a stack must obtain feedback from the user before it can continue.

If the stack is already open, the sheet command closes the stack and reopens it as a sheet, so closeStack and openStack, closeCard and openCard, and (if applicable) closeBackground and openBackground messages are sent to the parentStack's current card as a result of executing this command. Use the lock messages command before executing modal if you want to prevent the close messages from being sent; the open messages are sent regardless of the setting of the lockMessages property.

The sheet command pauses the running handler until the sheet is dismissed (usually by clicking a button in the sheet). To return information to the handler about which button was clicked, in the button's script, set a global variable or custom property. After the dialog box is dismissed, the handler can query this variable or property and act accordingly.

Attempting to open a sheet from within another sheet displays the second stack as a modal dialog box instead.

Cross-platform note: Sheets are only used on OS X systems. If you use the sheet command on a Mac OS, Unix, or Windows system, the stack is displayed as a modal dialog box and the parentStack parameter is ignored.

Changes to Transcript:

The ability to specify a parentStack was introduced in version 2.1. In previous versions, a sheet always appeared inside the defaultStack.

shell

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hideConsoleWindows property, launch command, MCISendString function, open process command, shellCommand property, sysError function

Summary

Runs a shell command and returns its output.

Syntax

the shell of commandLine

shell(commandLine)

Example Code

```
shell("ls -l *.txt") -- returns a listing of the current dir on Unix
shell(field "Command")
shell("attrib main.rev +R")
```

Comments

Use the shell function to execute a command line.

Parameters:

The commandLine is a string or an expression that evaluates to a string.

Value:

The shell function returns a string.

Comments:

The commandLine must be a valid shell command on the current operating system. Use the shellCommand property to set the shell you want to use.

Note: If you use a file path in the shell command on a Windows system, the file path must be in Windows form, not the Unix-style file paths that Revolution uses internally.

The value returned by the shell function is the result of the `commandLine`, including any error messages the `commandLine` generates. (On Unix systems, the `stdout` and `stderr` are combined to create the return value.) The current handler pauses until the shell returns its result. If the command was successful but did not return anything, the shell function returns empty.

The result is set to the shell command's exit code.

To prevent a console window from appearing when the shell command is executed, set the `hideConsoleWindows` property to `true`.

Tip: If you are having problems with the shell function, try executing the `commandLine` at your operating system's shell prompt. (For example, on an OS X system, try executing the `commandLine` in the Terminal window.) If a `commandLine` does not work at the shell prompt, it won't work with the shell function either, so trying this can be useful when debugging.

Changes to Transcript:

Support for using the shell command on OS X systems was added in version 2.0.

In version 2.0 and later, the shell function returns empty if a shell command (that does not normally return any data) was successful. In previous versions, the shell function returned zero in this case.

shellCommand

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hideConsoleWindows property, shell function

Summary

Specifies the name and location of the command used for the shell function.

Syntax

set the shellCommand to shellPath

Example Code

```
set the shellCommand to "/bin/sh/ksh"
```

Comments

Use the shellCommand property to execute a command with a particular shell.

Value:

The shellCommand is a string.

By default, the shellCommand property is set to `/bin/sh` (the Bourne shell) on Unix systems, and `command.com` on Windows systems.

Comments:

When you use the shell function, the command line you specify is sent to the program specified in the shellCommand.

On OS X and Unix systems, the shellCommand is the path to an executable file. On Windows systems, the shellCommand is a DOS command.

shiftKey

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

altKey function, commandKey function, controlKey function, down constant, keyDown message, keysDown function, keyUp message, metaKey function, optionKey function, up constant

Summary

Returns the state of the Shift key.

Syntax

the shiftKey
shiftKey()

Example Code

```
if shiftKey() is down then beep  
set the locked of me to (the shiftKey is up)
```

Comments

Use the shiftKey function to check whether the user is pressing the Shift key.

Value:

The shiftKey function returns down if the key is pressed and up if itÆs not.

short
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

abbreviated keyword, convert command, date function, ID property, long keyword, name property, time function, twelveHourTime property

Summary

Specifies a format for the date and time functions, the convert command, and the name, ID, and owner properties.

Syntax

Example Code

```
put the short name of this card into field "Card Label"
```

Comments

Use the short keyword to obtain a date, time, name, or ID in the most compressed form available.

Comments:

In general, a short form is shorter than either the abbreviated form or the long form.

If the useSystemDate property is set to false, a short date looks like this:

7/21/00

If the useSystemDate is true, the short date is formatted according to the system settings.

If the useSystemDate property is set to false, a short time looks like this:

11:22 AM

If the useSystemDate is true, the short time is formatted according to the system settings.

A short object name is simply its name, with no qualifiers. For example, a short card name looks like this: This Card

A short object ID looks like this: 2238

Note: The short keyword is implemented internally as a property and function, and appears in the `propertyNames` and the `functionNames`. However, it cannot be used as a property in an expression, nor with the `set` command.

Changes to Transcript:

The form the short owner was introduced in version 2.0. In previous versions, the form of the owner property could not be specified and reported only the abbreviated owner.

shortFilePath

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

filename property, longFilePath function, About filename specifications and file paths

Summary

Returns the 8.3-format equivalent of a Windows long file path.

Syntax

the shortFilePath of filePath

shortFilePath(filePath)

Example Code

```
the shortFilePath of myPath
```

```
if it is the shortFilePath of it then put it into myFilePath
```

Comments

Use the shortFilePath function to obtain the equivalent of a file path in short format for use in situations that require it.

Parameters:

The filePath is the location and name of the file or folder whose short equivalent you want. If you specify a name but not a location, Revolution assumes the file or folder is in the defaultFolder.

Value:

The shortFilePath function returns a file path.

Comments:

The shortFilePath function transforms the filePath without looking for the file, so it returns a value even if the filePath does not exist.

On Mac OS, OS X, or Unix systems, the shortFilePath simply returns the filePath without changing it.

show

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

hide command, show cards command, show groups command, show menubar command, show taskbar command, showInvisibles property, visible property, visual effect command, How to peek at invisible objects in a stack, Why don't visual effects work?, Recipe for a "roll credits" effect

Summary

Makes an object visible.

Syntax

show object [with visual [effect] effectName [speed] [to finalImage]]

Example Code

```
show stack "Help Palette"  
show image "Splash" with visual effect barn door open
```

Comments

Use the show command to show invisible objects.

Parameters:

The object is any open stack, or any control in an open stack.

The effectName, speed, and finalImage are described in the visual effect command. These three parameters operate the same way as they do with the visual effect command.

Comments:

Showing an object sets its visible property to true. You can also set the showInvisibles property to true to override the visible property and show all objects.

You can show a card without causing a script error, but doing so has no effect.

Showing a group shows all the controls in the group whose visible property is true.

Changes to Transcript:

The ability to use QuickTime special effects was introduced in version 1.1. In previous versions, only the built-in visual effects listed under the visual effect command were available.

show cards

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

go command, lockRecent property, lockScreen property

Summary

Flips through cards in the current stack.

Syntax

show {number | all} cards

Example Code

```
show 15 cards
```

```
show all cards
```

```
show (the number of card "Last") cards
```

Comments

Use the show cards command to do ôflipbookö-style animation or to display a number of cards quickly to the user.

Parameters:

The number is the number of cards to display.

Comments:

The show number form displays number cards, starting with the current card.

The show all form displays all the cards in the current stack, starting and ending with the current card.

show groups

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

hide groups command, link keyword, show command, show menubar command, underlineLinks property

Summary

Underlines grouped text.

Syntax

show groups

Example Code

```
show groups  
if the hilite of button "Show Hypertext" then show groups
```

Comments

Use the show groups command to display an underline under grouped text.

Comments:

Text with its textStyle set to `ôlinkô` is treated specially by the `clickText`, `clickChunk`, `mouseText`, and `mouseChunk` functions: a style run of grouped text is treated as a single word. This makes grouped text handy to use for hypertext or "clickable text" features.

The show groups command sets the global underlineLinks property to true. It does not affect the stack underlineLinks property, so if a stack's underlineLinks is false, show groups does not display the underlining in that stack.

show menubar

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

hide menubar command, show command, show taskbar command

Summary

Displays the menu bar on Mac OS systems.

Syntax

show menubar

Example Code

```
show menubar  
if the mouseV < 20 then show menubar
```

Comments

Use the show menubar command to display the menu bar if it has been hidden. For example, if the menu bar is hidden, you might want to show it only if the user moves the mouse to the top of the screen.

Comments:

The menu bar can be hidden only on Mac OS systems. This command has no effect on Windows or Unix systems.

Hiding the menu bar in Revolution does not affect its appearance in other applications; if the user switches to another application, the menu bar becomes visible. If the user switches back to Revolution, the menu bar is hidden again.

show taskbar

command

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

hide taskbar command, show command, show menubar command

Summary

Displays the task bar on Windows systems.

Syntax

show taskbar

Example Code

```
show taskbar  
if the controlKey is down then show taskbar
```

Comments

Use the show taskbar command to display the task bar if it has been hidden. For example, if the task bar is hidden, you might want to show it only if the user holds down the Control key.

Comments:

The task bar can be hidden only on Windows systems. This command has no effect on Mac OS, OS X, or Unix systems.

showBadge

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

play command, showController property, visible property

Summary

Specifies whether a player's badge icon is visible.

Syntax

set the showBadge of player to {true | false}

Example Code

```
set the showBadge of player "Example" to true
```

Comments

Use the showBadge property to let users show or hide the player controller bar.

Value:

The showBadge of a player is true or false.

By default, the showBadge property of newly created players is set to false.

Comments:

If the showBadge is true, a small icon appears in the lower left corner of the player when its showController property is false. Clicking the badge displays the controller bar. If the showBadge is false, the badge icon does not appear.

Set the showBadge property to true if you are setting the player's showController property to false, but want to display the controller bar if desired.

If the player's showController property is true, the setting of the showBadge property has no effect.

showBorder

property

Synonyms
showPen

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
borderColor property, borderPattern property, borderWidth property, bottomColor property, bottomPattern property, decorations property, printCardBorders property, shadow property, showFocusBorder property, threeD property, topColor property, topPattern property, How to give a border to a radio button cluster, Why is there a border around controls?

Summary
Specifies whether an outline is drawn around an object.

Syntax
set the showBorder of object to {true | false}

Example Code
set the showborder of group "Status" to true

Comments
Use the showBorder property to draw an outline around an object, or to make the object borderless.

Value:
The showBorder of an object is true or false.

Comments:
The appearance of the border depends on the object's threeD and borderWidth properties, and on the object type.

If the object's threeD property is set to false, the border is drawn using the borderColor property (or the borderPattern property, if it is not empty). If the object's threeD is true, the raised edges are drawn using the topColor or topPattern property, and the sunken edges are drawn using the bottomColor or bottomPattern.

If the object's showBorder property is set to false, and its backgroundColor is empty, its interior is drawn with the backgroundColor or backgroundPattern of its owner.

The borders of controls and groups follow the object's rectangle property. The border of a card is an outline just inside the boundary of the stack window. You can set the `showBorder` of a stack, but the setting has no effect.

showController

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

endTime property, looping property, play command, playLoudness property, playSelection property, playStopped message, showBadge property, showSelection property, start command, startTime property, stop command, Why doesn't the controller of a player respond to clicks?

Summary

Specifies whether a player's controller bar is visible.

Syntax

set the showController of player to {true | false}

Example Code

```
set the showController of player 4 to false
```

Comments

Use the showController property to enable access to the player's controls, or hide the controller bar.

Value:

The showController of a player is true or false.

By default, the showController property of newly created players is set to true.

Comments:

A player's controller bar is used to start, pause, rewind, and fast-forward a movie or sound, and to select a portion of a movie or sound for playback.

Set a player's showController property to false if you are providing these capabilities in the stack already, or if you do not want the user to be able to control playback of the movie or sound.

showFocusBorder

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

focus command, focusColor property, focusedObject function, focusIn message, focusOut message, showBorder property, traversalOn property

Summary

Displays a border around the active control.

Syntax

set the showFocusBorder of control to {true | false}

Example Code

```
set the showFocusBorder of field "Answer" to false
set the showFocusBorder of myControl to (the platform is not "MacOS")
```

Comments

Use the showFocusBorder property to turn off the standard (for Unix systems) focus border around the active control, or to show which field has the insertion point in a dialog box.

Value:

The showFocusBorder of an object is true or false.

By default, the showFocusBorder property of newly created objects is set to true.

Comments:

The appearance of the focus border is set by the focusColor and focusPattern properties.

If the lookAndFeel property is set to `ôMacintoshö` or `ôAppearance Managerö`, the setting of the showFocusBorder property affects only fields, images, and EPS objects. (To comply with user interface guidelines, this property should be set to true for fields in modal and modeless dialog boxes.)

If the lookAndFeel is set to `ôMotifö`, the showFocusBorder property affects all controls.

If the lookAndFeel is set to `ôWindows 95ö`, the showFocusBorder property has no effect.

The focusBorder of cards and stacks has no effect.

showHilite

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

autoArm property, autoHilite property, hilite property, showBorder property

Summary

Specifies whether the box or circle belonging to checkboxes and radio buttons is shown.

Syntax

set the showHilite of button to {true | false}

Example Code

```
set the showHilite of button 7 to true
```

Comments

Use the showHilite property for radio buttons and checkboxes in menus.

Value:

The showHilite of a button is true or false.

By default, the showHilite property of newly created buttons is set to false.

Comments:

If the showHilite of a radio button or checkbox button is true, the circle or square is displayed only if the button's hilite is true. If the showHilite of a button which is a menu item in a stack menu is true, the box or circle is never shown.

showIcon

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

armedIcon property, disabledIcon property, hilitedIcon property, icon property, visible property, visitedIcon property

Summary

Specifies whether a buttonÆs icon is visible.

Syntax

set the showIcon of button to {true | false}

Example Code

```
set the showIcon of button 3 to false
```

Comments

Use the showIcon property to change a buttonÆs appearance.

Value:

The showIcon of a button is true or false.

By default, the showIcon property of newly created buttons is set to false.

Comments:

If a buttonÆs showIcon property is set to true, the image specified in the buttonÆs icon property is displayed in the buttonÆs rectangle. If the showIcon is false, the icon is not displayed.

If the buttonÆs icon property is zero, the setting of this property has no effect.

A button with an icon may need to be resized to show the entire icon.

showInvisibles

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hide command, show command, visible property, View menu > Show Invisible Objects

Summary

Specifies whether invisible objects are shown anyway.

Syntax

set the showInvisibles to {true | false}

Example Code

```
set the showInvisibles to true
```

Comments

Use the showInvisibles property to override the visible property of objects and see all the objects on a card.

Value:

The showInvisibles is true or false.

By default, the showInvisibles property is set to false.

Comments:

Showing all invisible objects can make it easier to find a lost control, but can also make a layout confusing.

Invisible objects that are temporarily shown with the showInvisibles property respond to mouse clicks and key presses as though they were visible.

showLines

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

fixedLineHeight property, hGrid property, selectedText function, textHeight property, textShift property

Summary

Specifies whether dotted lines are shown on the text baselines of a field.

Syntax

set the showLines of field to {true | false}

Example Code

```
set the showLines of the target to true
```

Comments

Use the showLines property to control the appearance of a field.

Value:

The showLines of a field is true or false.

By default, the showLines property of newly created fields is set to false.

Comments:

If a field's showLines property is true, a dotted horizontal line is drawn across the field, every lineHeight pixels. The line falls at the text baseline, so descenders (in letters like ôpö and ôgö) extend into or below the line.

The dotted lines drawn by the showLines appear above the solid lines drawn by the hGrid property, and are drawn in the foregroundColor (or foregroundPattern).

If the field's fixedLineHeight property is set to false, the showLines property has no effect.

The line does not appear under text that is selected.

showName

property

Synonyms

Objects

button, graphic, group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

label property, margins property, name property, textFont property, textSize property, textStyle property, How to give a border to a radio button cluster, How to remove a window's title bar

Summary

Specifies whether an object's name is displayed within it.

Syntax

set the showName of {button | graphic | group} to {true | false}

Example Code

```
set the showName of button "Oops!" to true
```

Comments

Use the showName property to display a text label for an object.

Value:

The showName of a button, graphic, or group is true or false.

By default, the showName property of newly created buttons is set to true. The showName of newly created graphics and groups is set to false.

Comments:

If an object's showName property is set to true, its name is displayed inside its rectangle. If the object has a label property, the label is displayed instead of the name.

showPict

property

Synonyms

Objects

card

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

hide command, show command, visible property

Summary

This property is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the showPict of card to {true | false}

Example Code

Comments

In HyperCard, the showPict property determines whether the card image is visible.

In Revolution, setting this property does not cause a script error, but doing so has no effect.

showSelection

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

duration property, endTime property, play command, playSelection property, showBadge property, showController property, start command, startTime property, stop command, visible property

Summary

Specifies whether the user can select a portion of the movie or sound in a player.

Syntax

set the showSelection of player to {true | false}

Example Code

```
set the showSelection of player 3 to true
```

Comments

Use the showSelection property to play a portion of the movie or sound.

Value:

The showSelection of a player is true or false.

By default, the showSelection property of newly created players is set to false.

Comments:

If a player's showSelection property is set to true, the user can shift-drag the thumb on the player controller to set the player's startTime and endTime properties, and the controller bar indicates the selected section as a dark gray bar.

If the showController property is false, the setting of the showSelection has no effect.

showValue

property

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

endValue property, numberFormat property, startValue property, thumbPosition property

Summary

Specifies whether the current value of a scale-bar scrollbar is displayed in below the scrollbar.

Syntax

set the showValue of scrollbar to {true | false}

Example Code

```
set the showValue of scrollbar "Pages" to true
```

Comments

Use the showValue property to give the user an exact indication of the scrollbar's position.

Value:

The showValue of a scrollbar is true or false.

By default, the showValue property of newly created scrollbars is set to false.

Comments:

If the scrollbar's style property is set to `ôscrollbarö` or `ôprogressö`, the setting of its showValue property has no effect.

The showValue property displays the scale bar's current setting. If the scale bar is oriented vertically, the value is displayed to its right. If the scale bar is oriented horizontally, the value is displayed above (if the lookAndFeel is set to `ôMotifö`) or below it.

Whenever the scrollbar is scrolled and the thumb's position changes, the value is updated. (Use the startValue and endValue properties to set the scale of the scrollbar value.)

shutdown

message

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeBackground message, closeCard message, closeStack message, defaultStack property, quit command, shutdownRequest message, suspend message, signal message, startup message, How to respond to quitting an OS X application

Summary

Sent to the current stack when the application is quitting.

Syntax

shutdown

Example Code

```
on shutdown -- make sure a settings stack is saved before quitting
    save stack "Preferences"
    pass shutdown
end shutdown
```

Comments

Handle the shutdown message if you want to perform cleanup tasks before the application quits.

Comments:

The actual quit process is not triggered by the shutdown message, so trapping the message and not allowing it to pass does not prevent the application from quitting.

shutdownRequest

message

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

quit command, closeStackRequest message, shutdown message, How to respond to quitting an OS X application

Summary

Sent to the current stack when the user tries to quit the application.

Syntax

shutdownRequest

Example Code

```
on shutdownRequest -- confirm with the user:
  answer question "Are you sure you want to quit?" with "Yes" or "No"
  if it is "Yes" then pass shutdownRequest -- allow to quit
end shutdownRequest
```

Comments

Handle the shutdownRequest message if you want to prevent the user from quitting the application.

Comments:

If the shutdownRequest handler does not pass the message or send it to a further object in the message path, the application does not exit. Passing the message allows the application to quit.

On OS X and Unix systems, if the operating system sends a SIGTERM signal to the application, Revolution sends a shutdownRequest message to the current stack. Passing the message causes the application to quit normally. If the message is trapped, the signal is ignored.

Changes to Transcript:

Handling of the SIGTERM signal was introduced in version 2.0. In previous versions, the SIGTERM signal did not cause a message to be sent.

signal

message

Synonyms

Objects

card

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

appleEvent message, kill command, processID function, quit command, shutdown message

Summary

Sent to the current card when another process sends a kill signal to the application.

Syntax

signal sigNumber

Example Code

```
on signal -- let the user know:
    answer warning "Ack! I've been killed by another application!"
end signal
```

Comments

Handle the signal message to perform cleanup before an external process causes the application to exit.

Parameters:

The sigNumber is the number of the Unix kill signal (1 for SIGUSR1, 2 for SIGUSR2) that was sent to the application.

Comments:

The user can execute the Unix `kill` command to send a kill signal to the application. Other processes can use a Unix system call to send a kill signal.

Only signal 1 (SIGUSR1) and signal 2 (SIGUSR2) are supported. Other signals sent to the application do not cause the signal message to be sent.

Changes to Transcript:

Support for the signal message on OS X systems was added in version 2.0.

sin function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

asin function, cos function, pi constant, tan function

Summary

Returns the sine of an angle (in radians).

Syntax

the sin of angleInRadians
`sin(angleInRadians)`

Example Code

```
the sin of -pi/2 -- returns -1  
sin(the startAngle of graphic "Pointer")
```

Comments

Use the sin function to find the sine of an angle.

Parameters:

The angle is a positive or negative number, or an expression that evaluates to a number.

Value:

The sin function returns a number between -1 and 1.

Comments:

The sin function repeats every 2π radians:

$\sin(x) = \sin(x + 2 * \pi)$
for any value of x.

The sin function requires its input to be in radians. To provide an angle in degrees, use the following custom function:

```
function sinInDegrees angleInDegrees  
  return sin(angleInDegrees * pi / 180)
```

end sinInDegrees

six

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

busy constant, constant command, sixth keyword

Summary

Equivalent to the number 6.

Syntax

Example Code

```
beep six -- same as "beep 6"
```

Comments

Use the six constant when it is easier to read than the numeral 6.

sixth
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

last keyword, middle keyword, number property, six constant

Summary

Designates the sixth member of a set.

Syntax

Example Code

```
copy the sixth image  
subtract sixth line of me from first line of me
```

Comments

Use the sixth keyword in an object reference or chunk expression.

Comments:

The sixth keyword can be used to specify any object whose number property is 6. It can also be used to designate the sixth chunk in a chunk expression.

The word the is optional when using the sixth keyword.

size
property

Synonyms

Objects

image, EPS, audioClip, videoClip, stack

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

diskSpace function, hasMemory function, heapSpace function, How to determine the size of a file

Summary

Reports the amount of disk space taken by an object.

Syntax

get the size of object

Example Code

```
if the size of image myImage > 20000 then beep
```

Comments

Use the size property to judge how much memory an object takes when displayed.

Value:

The size property reports an integer.

This property is read-only and cannot be set.

Comments:

The size of an image, EPS object, audio clip, or video clip is the number of bytes of disk space that the object takes up.

The size property of a stack is always 10000. For stacks, this property is included in Transcript for compatibility with imported HyperCard stacks and does not report a meaningful value.

slash
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

backslash constant, constant command, About filename specifications and file paths

Summary

Equivalent to the / character (ASCII 47).

Syntax

Example Code

```
if last char of it is slash then put true into isAFolder
```

Comments

Use the slash constant as an easier-to-read replacement for "/".

slices

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

grid property, polySides property

Summary

Specifies how many angles a line can snap to when drawing a line or irregular polygon with the Shift key held down.

Syntax

set the slices to number

Example Code

```
set the slices to 4 -- only 90-degree angles with shift key  
set the slices to card field 2
```

Comments

Use the slices property to draw lines at specified angles.

Value:

The slices is an integer greater than 1.

By default, the slices property is 16.

Comments:

The slices property is specified in degrees. 360 (the number of degrees in a full circle) divided by the slices yields the angle of each slice to the next slice. For example, if the slices is set to 16, the angle of a line drawn with the Shift key held down can be any multiple of 22.5 degrees (360 divided by 16).

socketClosed

message

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close socket command, openSockets function, socketError message

Summary

Sent when the remote system closes a socket.

Syntax

socketClosed socketID

Example Code

```
on socketClosed theID -- close a progress window for the socket
  set the itemDelimiter to "|" -- get just the assigned name
  if there is a window (last item of theID)
    then close stack (last item of theID)
  end socketClosed
```

Comments

Handle the socketClosed message to perform cleanup after the remote system closes a socket.

Parameters:

The socketID is the identifier (set when you opened the socket) of the socket that's just been closed.

The socket identifier starts with the IP address of the host the socket is connected to, and may optionally include a port number (separated from the IP address by a colon). If there is more than one socket connected to that host and port, you can specify which socket by appending the connection name or number that was assigned when the socket was opened, separated from the port number by a vertical bar (|).

Comments:

The socketClosed message is sent to the object that opened the socket.

If the socket closed due to an error, the socketError message is sent instead of socketClosed.

socketError

message

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close socket command, open socket command, openSockets function, read from socket command, resetAll command, socketClosed message, socketTimeout message, write to socket command

Summary

Sent when an error occurs on a socket that causes the socket to close or to fail to open.

Syntax

socketError socketID,errorString

Example Code

```
on socketError theID, theError
  answer error "There is a problem with the connection."

  with "Debugging Info" or "Cancel"
    if it is "Debugging Info" then answer information theError
  close socket theID
end socketError
```

Comments

Handle the socketError message to perform cleanup after a socket closes due to a problem receiving or sending data.

Parameters:

The socketID is the identifier (set when you opened the socket) of the socket which had an error.

The socket identifier starts with the IP address of the host the socket is connected to, and may optionally include a port number (separated from the IP address by a colon). If there is more than one socket connected to that host and port, you can specify which socket by appending the connection name or number that was assigned when the socket was opened, separated from the port number by a vertical bar (|).

The errorString is a description of the socket error.

Comments:

The `socketError` message is sent to the object that opened the socket.

If the socket is closed by the remote host normally, rather than due to an error, the `socketClosed` message is sent instead of `socketError`.

socketTimeout

message

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close socket command, open socket command, openSockets function, read from socket command, socketError message, socketTimeoutInterval property, write to socket command

Summary

Sent when a read from socket or write to socket command halts for the time specified by the socketTimeoutInterval property.

Syntax

socketTimeout socketID

Example Code

```
on socketTimeout theID
  answer error "The connection is not responding."

  with "Keep Trying" or "Cancel"
    if it is "Cancel" then close socket theID
  end if
```

Comments

Handle the socketTimeout message to take action when a socket communication has not completed within the time set by the socketTimeoutInterval.

Parameters:

The socketID is the identifier (set when you opened the socket) of the socket which had an error.

The socket identifier starts with the IP address of the host the socket is connected to, and may optionally include a port number (separated from the IP address by a colon). If there is more than one socket connected to that host and port, you can specify which socket by appending the connection name or number that was assigned when the socket was opened, separated from the port number by a vertical bar (|).

Comments:

The socketTimeout message is sent to the object that attempted to read from or write to the socket.

socketTimeoutInterval

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

read from socket command, socketError message, socketTimeout message, write to socket command

Summary

Specifies how long to wait for a read from socket or write to socket command to finish before timing out.

Syntax

set the socketTimeoutInterval to milliseconds

Example Code

```
set the socketTimeoutInterval to 100 -- one-tenth of a second
```

Comments

Use the socketTimeoutInterval to specify a time to wait for socket actions to complete.

Value:

The socketTimeoutInterval is a positive integer.

By default, the socketTimeoutInterval is 10000 (ten seconds).

Comments:

If the socketTimeoutInterval passes and no data has been transmitted yet, the socketTimeout message is sent to the object whose script contains the read from socket or write to socket command.

Revolution checks the socketTimeoutInterval every time a read from socket or write to socket command is executed, so you can specify different intervals for different commands by changing the socketTimeoutInterval before issuing the command.

As long as the action is still pending, the socketTimeout message is sent every time the socketTimeoutInterval elapses. For example, if the socketTimeoutInterval is 1000 milliseconds (one second), a socketTimeout message is sent every second until the action completes.

sort

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

ascending keyword, convert command, dateTime keyword, descending keyword, find command, mark property, number property, numeric keyword, sort container command, text keyword, How to change the order of cards in a stack, How to shuffle the cards in a stack

Summary

Sorts the cards of a stack into a new order.

Syntax

sort [marked] cards [of stack] [direction] [sortType] by sortKey

Example Code

```
sort cards by field "Name"  
sort cards of this stack by random(the number of cards)  
sort marked cards of stack "Next" numeric by the number of buttons  
sort cards numeric by the short ID of this card  
sort cards of stack "Manson" descending dateTime by item 1 of field "Date"
```

Comments

Use the sort command to shuffle the cards in a stack into a new order, or to shuffle a subset of the cards in a stack.

Parameters:

The stack is a reference to any open stack. If you don't specify a stack, the cards of the current stack are sorted.

The direction is either ascending or descending. If you don't specify a direction, the sort is ascending.

The sortType is one of text, international, numeric, or dateTime. If you don't specify a sortType, the sort is by text.

The sortKey is an expression that evaluates to a value for each card in the stack. Any object references within the sortKey are treated as pertaining to each card being evaluated, so for example, a reference to a field is evaluated according to that field's contents on each card.

Comments:

The sort text form sorts by the ASCII value of each character.

The sort international form is like sort text, except that it sorts accented characters together with their unaccented counterparts. For example, ôÚö sorts between ôeö and ôfö, rather than at the end of the alphabet as it would using the sort text form.

The sort numeric form sorts by number. (Use this form if the sortKey consists of numbers.)

The sort dateTime form treats the sortKey as a date and/or time.

The sort command is a stable sort. This means that if the sortKey for two cards is the same, sorting does not change their order, so you can do two successive sorts to create subcategories within the major sort categories. For example, to sort the cards of a stack by ZIP code and sort within each ZIP code by last names, use these two statements:

```
sort cards by field "Last Name"  
sort cards numeric by field "ZIP code"
```

sort container

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

ascending keyword, convert command, descending keyword, offset function, sort command, How to shuffle the items or lines in a container, How to sort a portion of a container, Recipe for sorting a list of titles

Summary

Sorts the lines or items of a container into a new order.

Syntax

sort [{lines | items} of] container [direction] [sortType] [by sortKey]

Example Code

```
sort field "Output"  
sort items of myInfo by word 2 of each -- sort by word 2 of the line  
sort lines of field thisField descending numeric by item x of each
```

Comments

Use the sort container command to shuffle the order of lines or items in a container.

Parameters:

The container is a field, button, or variable, or the message box.

The direction is either ascending or descending. If you don't specify a direction, the sort is ascending.

The sortType is one of text, numeric, or dateTime. If you don't specify a sortType, the sort is by text.

The sortKey is an expression that evaluates to a value for each line or item in the container. If the sortKey contains a chunk expression, the keyword each indicates that the chunk expression is evaluated for each line or item. If you don't specify a sortKey, the entire line (or item) is used as the sortKey.

Comments:

If you don't specify lines or items, the lines of the container are sorted.

The each keyword, when used in the sortKey, specifies the entire line or item. You can use the each keyword in an expression, as a placeholder for the current line or item. For example, this statement sorts the lines of a variable by the third word of each line:

```
sort lines of myVariable by word 3 of each
```

You can use the each keyword in any expression, not just chunk expressions.

The sort text form sorts by the ASCII value of each character. The sort numeric form sorts by number. Use this form if the sortKey consists of numbers. The sort dateTime form treats the sortKey as a date and/or time.

The sort container command is a stable sort. This means that if the sortKey for two items or lines is the same, sorting does not change their order, so you can do two successive sorts to create subcategories within the major sort categories.

Tip: To create a custom sort order, use the each keyword to pass each line or item to a custom function. The value returned by the function is used as the sort key for that line or item.

sound

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

done constant, movie function, play command, recording property, revIsSpeaking function, templateAudioClip keyword

Summary

Returns the name of the audio clip thatÆs currently playing.

Syntax

the sound
sound()

Example Code

```
the sound  
wait until the sound is done
```

Comments

Use the sound function to synchronize other actions with sounds.

Value:

The sound function returns a string.

Comments:

If no audio clip is playing, the sound function returns done.

Sounds played with the beep command do not affect the value returned by the sound function.

soundChannel

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

play command, sound function

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the soundChannel to channelNumber

Example Code

Comments

In HyperCard, the soundChannel property determines which sound channel is used when Mac OS sound resources are played. In Revolution, this property has no effect.

By default, the soundChannel property is set to zero. A handler can set it to any number without causing a script error, but the change has no effect.

space
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

&& operator, constant command, tab constant

Summary

Equivalent to the space character (ASCII 32).

Syntax

Example Code

```
"a" & space & "b" -- yields "a b"  
put firstName & space & lastName into fullName
```

Comments

Use the space constant as an easier-to-read substitute for " ".

specialFolderPath

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

Note: this command works differently across platforms.

See Also:

aliasReference function, defaultFolder property, files function, folders function, tempName function, How to find the Preferences folder, How to store preferences or data for a standalone application

Summary

Returns the names and locations of system-related folders.

Syntax

the specialFolderPath of folderIdentifier

specialFolderPath(folderIdentifier)

Example Code

```
specialFolderPath("Preferences")
set the defaultFolder to specialFolderPath("documents")
put specialFolderPath("asup") into appSupportFolder -- special constant
```

Comments

Use the specialFolderPath function to place preferences in the Preferences folder, save files to the Documents folder, find out which files are installed in the System or Fonts folders, and so on.

Parameters:

The folderIdentifier is one of the following:

For Mac OS systems:

Desktop: The Desktop folder, where files or folders on the desktop are placed

System: The active System Folder

Apple: The Apple Menu Items folder inside the System Folder

Control: The Control Panels folder inside the System Folder

Extension: The Extensions folder inside the System Folder

Fonts: The Fonts folder inside the System Folder

Preferences: The Preferences folder inside the System Folder

Temporary: For storage of temporary files

For OS X systems:

Desktop: The user's Desktop folder

System: The active System Folder

Fonts: The system Fonts folder

Preferences: The Preferences folder inside the user's Library folder

For Mac OS and OS X systems, you can also specify a four-character special folder constant as the `folderIdentifier`. You can find a list of these constants on the Apple web site at

<http://developer.apple.com/techpubs/macosx/Carbon/Files/FolderManager/Folder_Manager/folder_manager_ref/constant_6.html#apple_ref/c/tdef/FolderType>.

Note: Some special folder constants are meaningful only on Mac OS or only on OS X but not both. Not all special folder constants work on all system versions.

For Windows systems:

Desktop: For icons on the desktop

System: For system components

Start: For items in the Start menu

Documents: For user documents

Fonts: For font storage

Temporary: For storage of temporary files

For Windows systems, you can also specify a CSIDL number that identifies a special folder. You can find a list of available CDISL values on the Microsoft web site at

<<http://msdn.microsoft.com/library/en-us/shellcc/platform/shell/reference/enums/csidl.asp>>.

Note: The CSIDL numbers are provided in hexadecimal notation, so you'll need to use the `baseConvert` function to translate them into decimal numbers to use them with the `specialFolderPath` function. For example, the CSIDL number of the "My Music" folder is hexadecimal 0x000d. The expression `baseConvert("0x000d",16,10)` is equal to 13, the number in decimal notation, so you use 13 as the `folderIdentifier` to obtain the location of the "My Music" folder.

Note: Not all CSIDL numbers work on all Windows versions.

Value:

The `specialFolderPath` function returns a folder name and location.

Comments:

If the folder is not found, the `specialFolderPath` function returns empty. If the `folderIdentifier` is either not supported on the current platform or doesn't exist on the current system, the result is set to "folder not found".

Some of the special folders you can specify with the `specialFolderPath` function need not have the standard names. (For example, the System Folder on Mac OS systems is not necessarily named "System Folder".) For this reason, the `specialFolderPath` function returns the full path to the folder, including the name of the folder.

Changes to Transcript:

Support for special folder constants (on Mac OS and OS X) and CSIDL numbers (on Windows) was added in version 2.0.

split

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

& operator, [] keyword, combine command, How to find out whether a container is an array

Summary

Transforms a list into an array.

Syntax

split variable {by|using|with} primaryDelimiter [and secondaryDelimiter]

Example Code

```
split it by return  
split currentRisks by tab and ";"
```

Comments

Use the split command to place a list in an array so you can easily address each part of the list.

Parameters:

The variable is any variable that is not an array.

The primaryDelimiter is a character whose ASCII value is in the range 1 to 127, or an expression that evaluates to such a character.

The secondaryDelimiter is a character (other than the primaryDelimiter) whose ASCII value is in the range 1 to 127, or an expression that evaluates to such a character.

Comments:

The split command separates the parts of the variable into elements of an array. After the command is finished executing, the variable specified is an array.

The parts that become elements are defined by the primaryDelimiter. For example, if the primaryDelimiter is return, each line of the variable becomes an element in the resulting array.

If you don't specify a secondaryDelimiter, then a simple numeric array is created, with each key being a number, starting with 1.

If you specify a secondaryDelimiter, the key for each element is the first portion of each part of the variable, separated from the element's content by the secondaryDelimiter. For example, if the primaryDelimiter is return and the secondaryDelimiter is space, the remainder of each line of the variable is placed in an array element whose key is the first word of the line.

For example, the following statements create an array:

```
put "A apple,B bottle,C cradle" into myVariable
split myVariable by comma and space
```

The resulting array looks like this:

KEY	VALUE
A	apple
B	bottle
C	cradle

Important! Using the split command can discard data if any of the keys in the original variable are duplicated. If more than one part of the variable delimited by the primaryDelimiter has the same first portion delimited by the secondaryDelimiter, only the element corresponding to the first part is created. (For example, if you are splitting a variable by return and space, and two lines happen to have the same first word, only one of the lines is retained in the array.) Only one element is created for each unique key.

spray

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

brush property, choose command, eraser property, spray can keyword, tool property

Summary

Specifies the shape used for painting with the Spray Can tool.

Syntax

set the spray to {brushID | imageID}

Example Code

```
set the spray to 30
```

Comments

Use the spray property to specify which shape is painted by the Spray Can tool.

Value:

The spray is a brush specifier.

A brushID is a built-in brush number between 1 and 100. (These brushes correspond to Revolution's built-in patterns 100 to 135.)

An imageID is the ID of an image to use for painting with the spray can. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the spray is set to 34 (a round splatter pattern).

Comments:

The entire painted area of the brush is used as the spray can shape. The shape painted by the spray can is drawn in the brushColor, regardless of what colors might be in the image used for the spray can shape.

When the Spray Can tool is in use, the cursor is the same as the spray can shape. You can use any size image as a spray can, but the cursor may appear distorted on some systems if the image is not 16x16 pixels.

spray can
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

brushColor property, brushPattern property, choose command, spray property, tool function

Summary

Designates the paint tool used to add airbrushed color to an image.

Syntax

Example Code

```
choose spray can tool
```

Comments

Use the spray can keyword to paint airbrush strokes with the brushColor.

Comments:

When using the Spray Can tool, the cursor is the shape specified by the spray property (over images) or an arrow (anywhere else). This tool is used to brush the brushColor (or brushPattern) onto an image in the pattern of the spray.

If you try to paint with the Spray Can tool on a card that has no images, an image object the size of the card is created to hold the painting. If the current card already has one or more images, painting outside the image has no effect.

sqrt

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

\wedge operator, exp function, ln function

Summary

Returns the square root of a number.

Syntax

the sqrt of number
`sqrt(number)`

Example Code

```
sqrt(9) -- returns 3  
put sqrt(sideA^2 + sideB^2) into hypotenuseLength
```

Comments

Use the sqrt function to find the square root of a number.

Parameters:

The number is any non-negative number, or an expression that evaluates to such a number.

Value:

The sqrt function returns a non-negative number.

Comments:

The square root of a number is the number which must be squared to obtain number: $\text{sqrt}(\text{number})^2$ is equal to number.

srcAnd

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of graphic "Outline" to srcAnd
```

Comments

Use the srcAnd keyword to make the light-colored parts of an object transparent.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

When the srcAnd transfer mode is used, Revolution performs a bitAnd operation on each component of the object color with the corresponding component of the color under the object.

Each component of the resulting color is equal to the result of this expression:

object component bitAnd background component

The effect is that the lighter an object is, the more transparent it is. White parts of an object are completely transparent, and black parts are completely opaque.

For example, suppose an object's color is 45,0,255, and the color of the pixels under the object is 20,45,100. If the srcAnd transfer mode is used, the object's displayed color is 4,0,100 (the decimal equivalent).

The srcAnd mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear as though their ink were set to noOp.

srcAndReverse

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of field "Check" to srcAndReverse
```

Comments

Use the srcAndReverse keyword to invert the color underneath the light-colored parts of an object.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

The srcAndReverse transfer mode performs the following steps to compute the color of each pixel:

1. Each component of the color underneath the object—the number indicating the amount of red, green, or blue—is changed to its inverse. (If the color is expressed as three integers between zero and 255—one for each of red, green, and blue—then the inverse of each component is equal to 255 minus the component's value.)
2. Revolution performs a bitAnd operation on each component of the inverted object color with the corresponding component of the color under the object.

Each component of the resulting color is equal to the result of this expression:
object component bitAnd (255 - background component)

The effect is that the lighter an object is, the more transparent it is to the inverted color beneath. White parts of an object completely invert the color underneath them, and black parts are completely opaque.

For example, suppose an object's color is 45,0,255, and the color of the pixels under the object is 20,45,100. The inverse of the color underneath is obtained by subtracting each component from 255, yielding 235,210,155. If the srcAndReverse transfer mode is used, the object's displayed color is 41,0,155 (the decimal equivalent).

The srcAndReverse mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear as though their ink were set to srcCopy.

srcBic

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of graphic "Square" to srcBic
```

Comments

Use the srcBic keyword to turn an object white.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed. When the srcBic mode is used, each component of the color underneath the object's red, green, and blue is raised to be equal to white.

The srcBic mode can be used only on Mac OS systems. On Unix and Windows systems, objects whose ink property is set to this mode appears as though their ink were set to srcCopy.

srcCopy

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of button "Move" to srcCopy
```

Comments

Use the srcCopy keyword to make an object opaque.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

When the srcCopy mode is used, each pixel of the object is drawn over the pixels underneath, blocking them completely. The color of the pixel underneath does not affect the final color.

srcOr
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of control 11 to srcOr
```

Comments

Use the srcOr keyword to make the dark-colored parts of an object transparent.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

When the srcOr transfer mode is used, Revolution performs a bitOr operation on each component of the object color with the corresponding component of the color under the object.

Each component of the resulting color is equal to the result of this expression:

object component bitOr background component

The effect is that the darker an object is, the more transparent it is. Black parts of an object are completely transparent, and white parts are completely opaque.

For example, suppose an object's color is 45,0,255, and the color of the pixels under the object is 20,45,100. If the srcOr transfer mode is used, the object's displayed color is 61,0,255 (the decimal equivalent).

The srcOr mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear as though their ink were set to srcCopy.

srcOrReverse

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcXOr keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of card field ID 8 to srcOrReverse
```

Comments

Use the srcOrReverse keyword to invert the color underneath the dark-colored parts of an object.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

The srcOrReverse transfer mode performs the following steps to compute the color of each pixel:

1. Each component of the color underneath the object—the number indicating the amount of red, green, or blue—is changed to its inverse. (If the color is expressed as three integers between zero and 255—one for each of red, green, and blue—then the inverse of each component is equal to 255 minus the component's value.)
2. For each of the three components, the resulting number is converted to its binary equivalent, a string of ones and zeroes. The same is done for each component of the object color.

3. Revolution performs a bitOr operation on each component of the object color with the corresponding component of the inverse color under the object. (A bit is 1 if one or both of the corresponding bits of the object color and the color underneath is 1. If both corresponding bits are zero, the resulting bit is 0.)
4. For each component, the resulting binary number is turned back into a decimal number, which is one of the three components (red, green, or blue) of the final color.

The effect is that the darker an object is, the more transparent it is to the inverted color beneath. Black parts of an object completely invert the color underneath them, and white parts are completely opaque.

For example, suppose an object's color is 45,0,255, and the color of the pixels under the object is 20,45,100. The binary equivalent of the object's color is

00101101, 00000000, 11111111

and the binary equivalent of the inverse of the color underneath is

11101011, 11010010, 10011011

If the srcOrReverse transfer mode is used, the resulting binary components are

11101111, 11010010, 11111111

and the object's displayed color is 239,210,255 (the decimal equivalent).

The srcOrReverse mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear as though their ink were set to srcCopy.

srcXOr

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, subOver keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of field "Meld" of group 1 to srcXOr
```

Comments

Use the srcXOr keyword to combine an object's colors with the color underneath the object.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed.

When the srcXOr transfer mode is used, Revolution performs a bitXOr operation on each component of the object color with the corresponding component of the color under the object.

Each component of the resulting color is equal to the result of this expression:

object component bitXOr background component

For example, suppose an object's color is 45,0,255, and the color of the pixels under the object is 20,45,100. If the srcXOr transfer mode is used, the object's displayed color is 57,45,210 (the decimal equivalent).

The srcXOr mode can be used only on Unix and Windows systems. On Mac OS systems, objects whose ink property is set to this mode appear as though their ink were set to reverse.

stack object

Synonyms

window, wd

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

defaultStack property, mode property, mouseStack function, revLoadedStacks function, templateStack keyword, topStack function, About object types and object references, File menu > New Mainstack, Tools menu > Application Browser

Summary

The basic Revolution object: a window.

Syntax

Example Code

```
go to stack "Help"  
put the short name of stack 3 after field "Stacks List"
```

Comments

Use the stack object type to display a window, palette, or dialog box.

Comments:

A stack corresponds to a single window. Each Revolution file contains one or more stacks. The first stack in a stack file is called the main stack; any other stacks in the stack file are called substacks of the main stack.

Stacks contain one or more cards, and may optionally contain backgrounds.

Note: The type of window a stack is displayed in depends on the stack's style property and whether the stack was opened with the go, topLevel, palette, modal, or modeless command.

The stack object has a number of properties and messages associated with it. To see a list of messages that can be sent to a stack as a result of user actions or internal Revolution events, open the "Transcript Language Dictionary" page of the main Documentation window, and choose "Stack Messages" from the Show menu at the top. To see a list of all the properties a stack can have, choose "Stack Properties" from the Show menu.

stackFiles

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

filename of stack property, mainStack property, mainStacks function, stackFileType property, there is a operator, About main stacks, substacks, and the organization of a stack file

Summary

Specifies a list of stacks and their file names, to be used when a stack thatÆs not open is referenced in a handler.

Syntax

set the stackFiles of stack to stackFileList

Example Code

```
set the stackFiles of this stack to "My Dialog,Custom Dialogs.rev"
```

Comments

Use the stackFiles property to make one or more stacks accessible to handlers, even if the stack is not already open or in memory.

Value:

The stackFiles of a stack is a list of stack references, one per line.

Each stack reference consists of the short name of a stack, a comma, and the name of the file that contains that stack. (Do not put a space after the comma, unless the file name starts with a space.)

By default, the stackFiles property of newly created stacks is set to empty.

Comments:

When a handler or object refers to an object in a stack other than the current stack, Revolution checks all stacks that are loaded into memory (and their substacks) to find the referenced stack. If the stack cannot be found, Revolution checks the current stackÆs stackFiles property to locate the stack being referenced, and loads it into memory so that its properties and the objects in it can be used.

For example, if your stack contains a statement like this, the stack `Customers` must either be loaded into memory, or be listed in the `stackFiles`, in order for Revolution to know where it is:

```
get the width of field "Text" of card "Data" of stack "Customers"
```

In this example, if the `Customers` stack is not open and isn't in the `stackFiles`, the statement must refer to it by its full long name, including its file path:

```
get the width of field "Text" of card "Data" of stack "Customers"  
of stack "/Volumes/Data/Business.rev"
```

Using the `stackFiles` lets you simply specify the stack's name, instead of including its file path everywhere you refer to an object in the stack.

The `stackFiles` of a main stack is inherited by its substacks: if a handler in a substack refers to a stack that's not loaded, Revolution checks the `stackFiles` of both the substack and its main stack.

Important! Relative file paths in the `stackFiles` start from the folder that the stack is in, rather than starting from the default folder as with other relative paths in Transcript.

Placing a stack in the `stackFiles` lets handlers in your stack refer to objects in the referenced stack, but does not automatically allow handlers in your stack to call handlers in the referenced stack's scripts. To use a stack as a library, allowing your handlers to call the stack's handlers, use the `start using` or `insert script` command.

stackFileType

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

create stack command, fileName property, fileType property, save command, How to assign a custom creator signature to a standalone, How to change the file type of an existing file

Summary

Specifies the creator signature and file type for saved stack files.

Syntax

set the stackFileType to creator & type

Example Code

```
set the stackFileType to "myapSTCK"
```

Comments

Use the stackFileType property to ensure that stacks that a standalone application creates are recognized by the operating system as belonging to the standalone.

Value:

The stackFileType is an eight-character string. The first four characters are the creator signature, and the last four are the file type.

By default, the stackFileType is set to `RevoRSTK`.

Comments:

When a file is saved on a Mac OS system, a 4-character file type and 4-character creator signature are saved with it. The creator signature specifies which application owns the file, and the file type is used to determine which applications (other than the owner) can work with the file.

Note: Revolution's creator signature is 'Revo'. The file type for a stack file created by Revolution is 'RSTK'.

The stackFileType property is used to set the file type and creator for stack files your application creates with the save command. (To specify the file type and creator for files created by the open file command

or by putting data into a file, binfile, or resfile URL that doesn't yet exist, use the fileType property instead.)

The setting of this property has no effect on Unix and Windows systems.

Important! To avoid conflicts with other applications, register any new creator signatures with Apple Computer if you plan to distribute a stack or standalone application that uses this property.

stacks

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

defaultStack property, name property, openStacks function, reloadStack message, revLoadedStacks function, topStack function, visible property, How to find out whether a window is open

Summary

Returns a list of stack files that contain an open stack.

Syntax

the stacks
stacks()

Example Code

```
the stacks
```

Comments

Use the stacks function to find out which stack files are open.

Value:

The stacks function returns a list of file paths of open stacks, one per line.

Comments:

Since unsaved stacks have no file name, the file names of unsaved stacks are not included in the list. Instead, each unsaved stack is represented by a blank line in the list.

The file paths of substacks are included, but the stack names themselves are not. This means that if more than one stack in a file is open, that file's path appears more than once in the list returned by the stacks function.

The list includes any open Revolution windows (such as the message box and menu bar). It also includes stacks that have been opened during the current session and then closed, unless the stack's destroyStack property is set to true.

stacksInUse

property

Synonyms
libraries

Objects
global

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
backScripts function, frontScripts function, releaseStack message, scriptLimits function, start using command, stop using command, How to create a code library

Summary
Reports the stacks that have been placed in the message path by the start using command.

Syntax
get the stacksInUse

Example Code
if the short name of this stack is not among the lines
 of the stacksInUse then start using this stack

Comments
Use the stacksInUse property to find out which stack scripts can intercept messages.

Value:
The stacksInUse property reports a list of stack names, one per line.

This property is read-only and cannot be set.

Comments:
The value reported by the stacksInUse property is a list of the short names of the stacks that have been added to the message path, one stack per line.

Add stacks to the message path with the start using command; remove them with the stop using command.

stackSpace

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

diskSpace function, hasMemory function, heapSpace function

Summary

Does not return a meaningful value and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

the stackSpace

stackSpace()

Example Code

Comments

In HyperCard, the stackSpace function returns the disk size of the current stack. In Revolution, this function does not return a meaningful value.

standard

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

style property, Object menu > New Control > Standard Button

Summary

Used with the style property to indicate a button that looks and behaves like a standard button for the current platform.

Syntax

Example Code

```
set the style of button "Cancel" to standard
```

Comments

Use the standard keyword to create a button that looks like a standard dialog-box button.

standardDeviation

function

Synonyms

stdDev

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

average function, max function, median function, min function, round function, sum function

Summary

Returns the standard deviation of a list of numbers.

Syntax

standardDeviation(numbersList)

Example Code

```
standardDeviation(8,10,12) -- returns 2  
put standardDeviation(studentScores) into field "Standard Deviation"
```

Comments

Use the standardDeviation function to analyze the amount of spread in a list of numbers.

Parameters:

The numbersList is a comma-separated list of numbers, or an expression that evaluates to such a list, or an array containing only numbers.

Value:

The standardDeviation function returns a number.

Comments:

The standard deviation is a measure of how widely distributed the numbers in the numbersList are.

start

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

currentTime property, movie function, paused property, play command, playStopped message, sound function, stop command, How to play a streaming QuickTime file

Summary

Resumes playing a paused movie or sound.

Syntax

start player

Example Code

```
start player "Birdcall Training"  
start player 1
```

Comments

Use the start command to start playing the movie or sound in a player.

Parameters:

The player is a player reference.

Comments:

The playing starts at the player's currentTime property.

start editing command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

editBackground property, group command, place command, remove command, stop editing command, ungroup command, Object menu > Edit Group

Summary

Puts a stack into group-editing mode.

Syntax

start editing group [of stack]

Example Code

```
start editing group "Expert"  
start editing group 1 of stack "Currencies"
```

Comments

Use the start editing command to add objects to a group or change the objects in a group.

Parameters:

The group is a reference to a group you want to edit.

The stack is a reference to the stack that contains the group you want to edit.

Comments:

If you don't specify a stack, the specified group in the current stack is put into group-editing mode.

Any objects created while in group-editing mode are added to that group.

When a group is in group-editing mode, font and color properties for the group (though not for objects in the group) are disregarded. This means that if properties such as textFont, foregroundColor, and so on are set for the group, it displays differently when in group-editing mode, using the fonts and colors of the stack the group is in. As soon as you exit group-editing mode with the stop editing command, the group's correct appearance is restored.

Note: The start editing command temporarily modifies the object hierarchy, displaying only the objects that belong to the group being edited. Inherited font and color properties may not appear correctly while the group is being edited, but will be restored when you exit group-editing mode. Also, if the start editing command is included in a handler, using the debugger to step through that handler may cause unexpected results because of the change in object context.

start using command

Synonyms
library

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

insert script command, libraryStack message, scriptLimits function, stacksInUse property, stop using command

Summary

Places a stack into the message path.

Syntax

start using stack

Example Code

```
start using stack "Random Commands"  
start using stack (field "Current Library")
```

Comments

Use the start using command to use a stackÆs script as a library for frequently-used handlers.

Parameters:

The stack is any stack reference.

Comments:

The start using command places a stackÆs script into the message path after the current stack and before any objects in the backScripts.

When you start using a stack, the libraryStack message is sent to the stack.

When a standalone application is running, it can insert up to fifty stacks into the stacksInUse. This limit is set by line 3 of the scriptLimits function. When using the development environment, you can insert any number of stacks into the message path.

startAngle

property

Synonyms

Objects

global, graphic

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

angle property, arcAngle property

Summary

Determines the point at which an arc begins.

Syntax

set the startAngle [of graphic] to angleInDegrees

Example Code

```
set the startAngle of graphic ID 673 to 90 -- top
set the startAngle to the startAngle + 12
```

Comments

Use the startAngle property to create an arc shape from an oval graphic, or to cause the Oval paint tool to draw arcs.

Value:

The startAngle of a graphic is an integer between zero and 359.

By default, the startAngle property of a newly created graphic is zero.

Comments:

By default, ovals display their entire arc from zero to 360°, forming a complete oval. Use the startAngle and arcAngle properties to specify that only a portion of the oval, forming an arc, should be drawn.

The startAngle determines the starting point of the arc. The angleInDegrees zero is at the right edge, 3 o'clock. Increasing the angleInDegrees moves the starting point counter-clockwise around the arc.

The global setting of the arcAngle property controls the appearance of arcs drawn with the paint tools. Once a paint arc is drawn, its appearance cannot be changed by changing the global arcAngle property.

For a graphic oval, the angle is the same as the startAngle. Changing one changes the other.

startArrow

property

Synonyms

Objects

graphic

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

arrowSize property, endArrow property, points property

Summary

Specifies whether a line graphic or the first vertex in an irregular polygon graphic has an arrowhead.

Syntax

set the startArrow of graphic to {true | false}

Example Code

```
set the startArrow of graphic "Pointer" to true
```

Comments

Use the startArrow property to place an arrowhead at the starting point of an irregular polygon or line graphic.

Value:

The startArrow of a graphic is true or false.

By default, the startArrow property of newly created graphics is set to false.

Comments:

For irregular polygons, the start arrow is placed at the point corresponding to the first line of the graphic/Es points property.

If the style property of the graphic is not polygon or line, the setting of its startArrow property has no effect.

startFrame

property

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

currentFrame property, endFrame property, frameCount property

Summary

The startFrame property is not implemented and is reserved.

Syntax

Example Code

Comments

startTime

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

duration property, endTime property, playRate property, playSelection property, selectionChanged message, showSelection property

Summary

Specifies the beginning of the selected portion of a sound or movie.

Syntax

set the startTime of player to startPoint

Example Code

```
set the startTime of player "Splash" to zero
```

Comments

Use the startTime property to set or get the selection in a player.

Value:

The startTime of a player is an integer between zero and the player's duration.

By default, the startTime property of newly created players is set to empty.

Comments:

If the playSelection property is true, only the selected portion plays, so you can use the startTime and endTime properties of the player to play whatever portion of the sound or movie you want.

The startTime is the number of the interval where the selection begins. (The number of intervals per second is specified by the player's timeScale property. The total number of intervals is given in the player's duration property.) If there is no selection, the startTime is empty.

startup

message

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

openBackground message, openCard message, openStack message, resume message, shutdown message, How to do a task when starting up the application, How to open a stack automatically when Revolution starts up

Summary

Sent to the first stack opened when the application starts up.

Syntax

startup

Example Code

```
on startup -- load prefs from a file in the same folder as the app
  global prefsSettings
  put URL "file:MyApp Preferences" into prefsSettings
  pass startup
end startup
```

Comments

Handle the startup message if you want to do initialization or other tasks when the application starts up.

Comments:

The startup message is sent only when the application starts up, not when a stack is opened when the application is already running.

If the application is opened with multiple stacks, the startup message is sent to the first stack opened.

startUpIconic

property

Synonyms

Objects

stack

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

icon property, iconic property, openStack message, topLevel command

Summary

Specifies whether, when a stack opens, it should be iconified or displayed as a window.

Syntax

set the startupIconic of stack to {true | false}

Example Code

```
set the startupIconic of stack "Imports" to true
```

Comments

Use the startUpIconic property to control a stack's appearance when it first opens.

Value:

The startUpIconic of a stack is true or false.

By default, the startUpIconic property of newly created stacks is set to false.

Comments:

If a stack's startUpIconic property is set to true, the stack is opened as a desktop icon rather than a window.

The setting of this property has no effect on Mac OS or Windows systems, nor on some Unix systems.

startValue

property

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

endValue property, showValue property, thumbPosition property

Summary

Specifies the value corresponding to a scrollbar's start position.

Syntax

set the startValue of scrollbar to number

Example Code

```
set the startValue of scrollbar "Progress" to 1
```

Comments

Use the startValue and endValue properties to set the scale of the scrollbar.

Value:

The startValue of a scrollbar is an integer between zero and 65535.

By default, the startValue property of newly created scrollbars is set to zero.

Comments:

The startValue is the value of the thumbPosition when the scrollbar thumb is all the way to the top (for a vertical scrollbar) or left (for a horizontal scrollbar).

statRound

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

average function, numberFormat property, round function, trunc function

Summary

Rounds off a number.

Syntax

the statRound of number

statRound(number,precision)

Example Code

```
the statRound of 2.5 -- yields 2, because 2 is even (rounds down)
```

```
the statRound of 3.5 -- yields 4, because 3 is odd (rounds up)
```

```
statRound(22.753, 2) -- yields 22.75
```

```
statRound(723.2, -1) -- yields 720
```

Comments

Use the statRound function to round off numbers and eliminate insignificant digits for statistical applications.

Parameters:

The number is any number or expression that evaluates to a number.

The precision is an integer giving the decimal place to round off to.

Value:

The statRound function returns a number.

Comments:

The statRound function performs statistical rounding. If the number is exactly halfway between two numbers, statRound rounds the number to the nearest even integer. (To round off numbers such numbers upward, as is the convention for financial applications, use the round function instead.) Because of this, a set of numbers rounded with statRound is not skewed upward, the way it is with the round function. Therefore, you should use statRound when statistical accuracy is required.

A positive precision indicates a place to the right of the decimal point, and a negative precision indicates a place to the left. For example, 1 rounds off to the nearest tenth, 2 rounds off to the nearest hundredth, -1 rounds off by ten, and so on. If you don't specify a precision, zero is used, meaning that the number is rounded off to a whole number.

Note: The statRound function is equivalent to HyperTalk's `round` function.

stderr
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems
Supported on Mac OS X systems
Not supported on Windows systems
Supported on UNIX systems

See Also:

EOF constant, stdin keyword, stderr keyword, write to file command, write to process command

Summary

Used with the write to file command to designate the standard error location.

Syntax

Example Code

```
write field 3 to stdout
```

Comments

Use the stderr keyword to write error messages to the standard error location (usually the screen or console window).

Comments:

On Unix systems, error messages from a program are normally displayed on the console, but can be redirected to go to another device, process, or file. This redirection is done on the command line when the program is started up.

When you use the write to file command with the stderr keyword, Revolution writes data to the standard error location.

Changes to Transcript:

Support for using stderr on OS X systems was added in version 2.0.

stdin

keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

EOF constant, read from file command, read from process command, stdout keyword

Summary

Used with the read from file command to designate the standard input.

Syntax

Example Code

```
read from stdin until EOF
```

Comments

Use the stdin keyword to read data from the standard input (usually the keyboard).

Comments:

On Unix systems, input to a program is normally read from the keyboard, but can be redirected to come from another device, process, or file. This redirection is done on the command line when the program is started up.

When you use the read from file command with the stdin keyword, Revolution reads data from the standard input.

Changes to Transcript:

Support for using stdin on OS X systems was added in version 2.0.

stdout
keyword

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

EOF constant, stdin keyword, stderr keyword, write to file command, write to process command

Summary

Used with the write to file command to designate the standard output.

Syntax

Example Code

```
write field 3 to stdout
```

Comments

Use the stdout keyword to write data to the standard output (usually the screen or console window).

Comments:

On Unix systems, output from a program is normally written to the screen, but can be redirected to go to another device, process, or file. This redirection is done on the command line when the program is started up.

When you use the write to file command with the stdout keyword, Revolution writes data to the standard output.

Note: On OS X systems, the standard output is the Console window—not the Terminal window.

Changes to Transcript:

Support for using stdout on OS X systems was added in version 2.0.

stop

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

currentTime property, movie function, play command, playDestination property, playStopped message, revGoToFramePaused command, revStopAnimation command, revStopPreviewingVideo command, revStopRecordingVideo command, sound function, start command

Summary

Stops a currently playing sound or movie.

Syntax

stop [playing] {player | videoClip | audioClip}

Example Code

```
stop player "Demo"  
stop playing audioClip "Trust No One"
```

Comments

Use the stop command to halt a sound or movie before it finishes. For example, you can use the stop command in the script of a button, to let the user click to stop a repeating sound.

Parameters:

The player is a reference to a player that holds a currently playing sound or movie.

The audioClip is a reference to an audio clip that is currently playing.

The videoClip is a reference to a video clip that is currently playing.

Comments:

The stop command stops playing a video clip or audio clip started with the play command.

stop editing

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

group command, place command, remove command, start editing command, ungroup command, Object menu > Stop Editing Group

Summary

Takes a stack out of group-editing mode.

Syntax

stop editing [group [of stack]]

Example Code

```
stop editing
stop editing background "Ideas" of stack "Idea Stack"
```

Comments

Use the stop editing command to stop editing a group.

Parameters:

The group is a reference to the group being edited.

Important! You must refer to the group using the background keyword.

The stack is a reference to the stack that contains the specified group.

Comments:

If you don't specify a group, the group currently being edited in the defaultStack is taken out of group-editing mode.

Any objects created after the stop editing command is executed are added to the current card, not a group.

stop moving

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

location property, lockMoves property, move command, moveStopped message, movingControls function

Summary

Stops an object that was set in motion by the move command.

Syntax

stop moving object

Example Code

```
stop moving card button 2  
stop moving the defaultStack
```

Comments

Use the stop moving command to halt an object before the move command completes.

Parameters:

The object is any window or control being moved with the move command.

Comments:

You can stop moving an object that is not in motion without causing an error, but doing so has no effect.

If you issue another move command while a previous move command for the same object is still executing, the previous move command is halted, and the second move command starts up at the object's current location.

To move multiple objects at the same time, set the lockMoves property to true before issuing the move commands. When you set the lockMoves back to false, all the pending moves begin at once.

stop recording

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

record sound command, recording property, revStopRecordingVideo command, stop command

Summary

Stops the current sound recording.

Syntax

stop recording

Example Code

```
if the mouse is down then stop recording
```

Comments

Use the stop recording command to halt a recording being made. For example, you can use the stop recording command in the script of a button, to let the user click to stop recording sound.

Comments:

The stop recording command stops a recording started with the record sound command.

The stop recording command sets the recording property to false.

stop using command

Synonyms

release library

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

releaseStack message, remove script command, stacksInUse property, start using command

Summary

Removes a stack from the message path that was placed in it with the start using command.

Syntax

stop using stack

Example Code

```
stop using stack "Network Functions Library"  
stop using stack myStackName
```

Comments

Use the stop using command to remove a stackÆs script from the message path.

Parameters:

The stack is any stack reference.

Comments:

You can stop using a stack only if it has previously been inserted in the message path with the start using command. The stop using command does not affect the current stack, which is normally in the message path, or scripts that were placed in the message path with the insert script command.

When you stop using a stack, the releaseStack message is sent to the stack.

strikeout

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bold keyword, italic keyword, plain keyword, textStyle property, underline keyword, Text menu > Strikeout

Summary

Used with the textStyle property to draw text with a horizontal line through it.

Syntax

Example Code

```
set the textStyle of the selectedChunk to strikeout
```

Comments

Use the strikeout keyword to indicate that text has been removed or superceded.

Comments:

You can apply the strikeout style to an object (striking out all the text displayed by the object) or a chunk in a field or button.

string
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

characters keyword, find command, normal keyword, whole keyword, words keyword

Summary

Used with the find command to search for a string of characters without regard to whether they occur at the start of a word.

Syntax

Example Code

```
find string "Help"  
find string "run" -- finds "run", "grunt", or "running"
```

Comments

Use the string keyword to find a string.

Comments:

When used with the find command, the string keyword finds cards that contain the specified string, whether each word in it is found as an entire word or not.

style property

Synonyms

Objects

button, field, graphic, scrollbar, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

hilite property, menuMode property, mode property, opaque property, shadow property, showBorder property, textStyle property, threeD property, About windows, palettes, and dialog boxes, How to give a border to a radio button cluster, How to make a throbbing button, Why can't I display a stack as a normal window?

Summary

Specifies the general appearance and behavior of an object.

Syntax

set the style of object to style

Example Code

```
set the style of card button 3 to rectangle
```

Comments

Use the style property to set an object's specific type.

Value:

The style of an object is a string.

Comments:

The possible settings of an object's style property depend on the kind of object.

The style of a button is one of the following:

- ò standard: the standard button for the current lookAndFeel setting
- ò transparent: no border; name is displayed but background is transparent
- ò opaque: background is the opaque backgroundColor of the button
- ò rectangle: opaque rectangular or square button with a border
- ò roundRect: opaque rectangular or square button with rounded corners
- ò shadow: opaque rectangular or square button with a drop shadow
- ò menu: a menu whose appearance is set by the menuMode property
- ò checkbox: a checkbox option button

ò radioButton: a radio button

The style of a button interacts with certain other properties. For example, setting the opaque of a standard button to false sets its style to ôtransparentö.

The style of a field is one of the following:

- ò transparent: no border; text is displayed but background is transparent
- ò opaque: background is the opaque backgroundColor of the field
- ò rectangle: opaque field with a border
- ò shadow: opaque field with a drop shadow
- ò scrolling: opaque field with a vertical scrollbar

The style of a graphic determines its shape and is one of the following:

- ò curve: a curved line
- ò line: a straight line
- ò oval: an oval or circle shape
- ò polygon: an irregular polygon shape
- ò rectangle: a rectangle or square shape
- ò regular: a regular polygon shape
- ò roundRect: a rectangle or square shape with rounded corners

The style of a scrollbar is one of the following:

- ò scrollbar: a standard scrollbar
- ò scale: a ôbead on a wireö slider control
- ò progress: a sliding progress bar display

Vertical scale and progress scrollbars always reflect the current lookAndFeel setting. Horizontal scale and progress scrollbars use the Motif look and feel, unless the lookAndFeel is set to ôAppearance Managerö.

The style of a stack determines its behavior and is one of the following:

- ò topLevel: editable window
- ò modeless: modeless dialog box, cannot be edited, can use only Browse tool
- ò modal: modal dialog box, can use only Browse tool
- ò palette: palette window, cannot be resized or edited, can use only Browse tool

subOver

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subPin keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of last card button to subOver
```

Comments

Use the subOver keyword to combine an object's color with the colors underneath it.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed. When the addOver mode is used, each component of the color underneath the object is subtracted from the corresponding component of the object's color. If the result is less than zero, the component rolls over backward, back to 255.

For example, suppose an object's color is 25,250,150, and the color of the pixels under the object is 60,40,100. If the subOver mode is used, the object's displayed color is 220,210,50.

The subOver mode can be used only on Mac OS systems. On Unix and Windows systems, objects whose ink property is set to this mode appears as though their ink were set to srcCopy.

subPin

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, transparent keyword

Summary

Specifies one of the transfer modes that can be used with the ink property.

Syntax

Example Code

```
set the ink of field 3 to subPin
```

Comments

Use the subPin keyword to combine an object's color with the colors underneath it.

Comments:

The ink property determines how an object's colors combine with the colors of the pixels underneath the object to determine how the object's color is displayed. When the subPin mode is used, each component of the object color (red, green, and blue) is subtracted from the corresponding component of the color underneath. If the resulting number is less than 127 (the minimum), 127 is used for that component.

For example, suppose an object's color is 240,0,100, and the color of the pixels under the object is 60,40,20. If the subPin mode is used, the object's displayed color is 180,127,127.

The subPin mode can be used only on Mac OS systems. On Unix and Windows systems, objects whose ink property is set to this mode appears as though their ink were set to srcCopy.

substacks

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

mainStack property, mainStacks function, owner property, stackFiles property, topStack function, About main stacks, substacks, and the organization of a stack file

Summary

Specifies which stacks are associated with a main stack, and saved in the same file.

Syntax

set the substacks of stack to substacksList

Example Code

```
put the substacks of this stack after loadedStacks
if the substacks of stack "Hello" contains "Goodbye" then exit mouseUp
get the substacks of stack "/Disk/Folder/stackfile.rev"
```

Comments

Use the substacks property to organize stacks in files.

Value:

The substacks of a stack is a list of stacks, one per line.

Comments:

Each Revolution file contains either a single main stack, or a main stack and one or more substacks. The substacks property reports on the substacks of a main stack.

The substacks of a substack is empty. Attempting to set the substacks property of a substack causes an execution error.

Setting the substacks of a main stack to empty removes all the substacks from memory. The next time the main stack is saved, the substacks are removed permanently.

Caution! You can move a substack from one main stack to another by setting the substacks property of the destination stack. However, since this may cause a conflict if more than one substack with the same name is open or in memory, it is safer to move a substack by setting the substack's mainStack property.

subtract

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

- operator, add command, divide command, multiply command, numberFormat property

Summary

Subtracts a number from a container and places the resulting value in the container.

Syntax

subtract number from [chunk of] container
subtract {number | array} from arrayContainer

Example Code

```
subtract 15 from oldNumber  
subtract field "Input" from word 2 of field "Output"  
subtract monthlyExpenses from monthlyTotals
```

Comments

Use the subtract command to subtract a number from a container or a portion of a container, or to subtract two arrays containing numbers.

Parameters:

The chunk is a chunk expression specifying a portion of the container.

The container is a field, button, or variable, or the message box.

The number is any expression that evaluates to a number.

The arrayContainer is an array variable each of whose elements is a number.

Comments:

The contents of the container (or the chunk of the container) must be a number or an expression that evaluates to a number.

If a number is subtracted from an arrayContainer, the number is subtracted from each element. If an array is subtracted from an arrayContainer, both arrays must have the same number of elements and the same dimension, and each element in the array is subtracted from the corresponding element of the arrayContainer.

If the container or an element of the arrayContainer is empty, the subtract command treats its contents as zero.

If container is a field or button, the format of the sum is determined by the numberFormat property.

Changes to Transcript:

The subtract from arrayContainer form was introduced in version 1.1. In previous versions, only single numbers could be used with the subtract command.

sum

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

+ operator, add command, max function, min function

Summary

Returns the sum of a list of numbers.

Syntax

sum(numbersList)

Example Code

```
sum(1,5,2,3) -- returns 11
```

Comments

Use the sum function to add a group of values together.

Parameters:

The numbersList is a comma-separated list of numbers, or an expression that evaluates to such a list, or an array containing only numbers.

Value:

The sum function returns a number.

Comments:

The sum function is equivalent to the following statements:

```
repeat with x = 1 to the number of items of numbersList
  add item x of numbersList to sumValue
end repeat
return sumValue
```

If the numbersList is empty, the sum function returns zero.

Changes to Transcript:

The ability to use an array was introduced in version 1.1. In previous versions, only lists of numbers could be used with the sum function.

surround

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

intersect keyword, select command, selection keyword, selectionMode property

Summary

Used with the selectionMode property to cause objects to be selected only when the entire object is within the rectangle you drag.

Syntax

Example Code

```
set the selectionMode to surround
```

Comments

Use the surround keyword to change the way selecting behaves.

Comments:

The surround keyword applies only to selections made with the Pointer tool. It does not affect text selections in a field.

suspend

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

closeStack message, focusOut message, shutdown message, suspendStack message, resume message

Summary

Sent to the current card when the application moves to the background.

Syntax

suspend

Example Code

```
on suspend
  hide stack "Toolbar"
end suspend
```

Comments

Handle the suspend message if you want to perform some action when the application is made inactive.

Comments:

The suspend message is sent whenever the user switches to another program.

The actual switch is not triggered by the suspend message, so trapping the message and not allowing it to pass does not prevent the user from switching out of the application.

suspendStack

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeStack message, focusOut message, iconifyStack message, resumeStack message, suspend message

Summary

Sent to the current card when something makes its stack no longer the active window.

Syntax

suspendStack

Example Code

```
on suspendStack -- hide a palette that only applies to this window
  hide stack "Accessories"
end suspendStack
```

Comments

Handle the suspendStack message if you want to perform some action when a stack window is made inactive.

Comments:

The suspendStack message is sent whenever a stack window is no longer the active window: when the stack window is closed, when another stack is brought to the front, when other Revolution windows are brought to the front, and when switching to another program.

The actual window deactivation process is not triggered by the suspendStack message, so trapping the message and not allowing it to pass does not prevent the stack window from becoming inactive.

switch

control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

break control structure, case keyword, default keyword, end switch keyword, if control structure, Recipe for a Cards menu

Summary

Chooses among several possible values for an expression, and executes a set of statements that depends on the value.

Syntax

```
switch [switchExpression]
  case {caseValue | caseCondition}
    [statementList]
  [default
    defaultStatementList]
end switch
```

Example Code

Comments

Use the switch control structure to select among multiple possible conditions, performing a different set of actions for each possibility.

Form:

The switch control structure begins with the word switch on a single line, with an optional switchExpression.

The switch line is followed by one or more case sections. Each case section begins with the case keyword, followed by either a caseValue (if a switchExpression was included on the switch line) or a caseCondition (if no switchExpression was included). If the caseValue is equal to the switchExpression, or the caseCondition evaluates to true, Revolution begins executing the following statements.

The case sections may be followed by an optional default section. If no break statement has been encountered yet in the switch control structure, the statements in the default section are executed.

The switch structure ends with an end switch statement.

Parameters:

The switchExpression is any expression.

The caseValue is any expression. (If the caseValue evaluates to the same value as the switchExpression, the condition is matched for that case section.)

The caseCondition is any expression that evaluates to true or false. (If the caseCondition evaluates to true, the condition is matched for that case section.)

Each statementList consists of one or more Transcript statements, and can also include if, switch, try, or repeat control structures.

The defaultStatementList consists of one or more Transcript statements.

Comments:

Flow of control in a switch structure is less complicated than it looks. In general, when Revolution enters a switch control structure, it looks for the first case section whose caseValue is equal to the switchExpression, or whose caseCondition is true. When a matching condition is found, all statements following it are executed—even statements in another case section—until either a break statement is encountered or the switch control structure ends.

This means that if you do not end a case section's statementList with a break statement, the statements in all the following case sections (and the default section) are executed even if those case sections don't have a matching caseValue or a true caseCondition. Occasionally, this behavior is useful. However, in most cases, you should place a break statement at the end of each statementList. This ensures that only one statementList is executed, and the rest are skipped.

This also means that you can attach more than one caseValue or caseCondition to the same statementList, simply by placing one case line above the next. The following example beeps if the current card is either the last or first card, and goes to the next card otherwise:

```
switch (the number of this card)
  case 1
  case (the number of cards)
    -- both the above case conditions execute the following
    -- statements:
    beep
    break
  default
    go next card
end switch
```


There is no limit to the number of case sections you can include in a switch control structure, although the more case sections there are, the more expressions Revolution must evaluate and the more slowly the switch structure executes.

Note: The switch control structure is implemented internally as a command and appears in the `commandNames`.

syncRate

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

blinkRate property, drag command, effectRate property, move command, visual effect command

Summary

Specifies how often the display is updated during visual effect, drag, and move commands.

Syntax

set the syncRate to number

Example Code

```
set the syncRate to 12
```

Comments

Use the syncRate property to specify how often the screen is redrawn.

Value:

The syncRate is a positive integer.

Comments:

Decreasing the syncRate reduces the load on the system, but may make the display of movements and visual effects more jerky.

sysError

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

errorDialog message, kill command, platform function, result function

Summary

Returns the operating system's error status.

Syntax

the sysError
sysError()

Example Code

```
the sysError  
if the sysError is zero then answer "Success!"
```

Comments

Use the sysError function to obtain the reason a command that makes an operating-system request failed.

Value:

The sysError function returns an integer.

Comments:

Most operating system requests have to do with actions on files. For example, when you open a file or copy a resource, Revolution requests the operating system to complete the action. If the action fails, the result is set to an error message, and the sysError function contains the error message the operating system reported to Revolution.

On Unix systems, the sysError function returns the value of the `errno` variable.

On Windows systems, the sysError function returns the value returned by the `GetLastError()` function.

Note: When using the sysError function to check whether a command succeeded, be sure to check the result first. The sysError reports the operating system's error report, and some operating-system

commands may report a value (and therefore set the `sysError`) even if the command succeeded. Only if the result is not empty does the `sysError` indicate that the command failed.

system
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

convert command, date function, english keyword, time function, useSystemDate property

Summary

Specifies that the date and time functions should be formatted in accordance with the user's system preferences.

Syntax

Example Code

```
put the system date && the system time into field "Clock"
```

Comments

Use the system keyword to correctly format dates and times that will be viewed by the user (for example, dates and times that are displayed in a field).

Comments:

The system format is set by the Date & Time control panel (on Mac OS systems), the Date control panel (on Windows systems), or the LANG environment variable (on Unix systems).

You can use the system keyword in combination with the long, abbreviated, or short keywords.

Note: The system keyword is implemented internally as a property and appears in the propertyNames. However, it cannot be used as a property in an expression, nor with the set command.

systemColorSelector

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

answer color command

Summary

The systemColorSelector property is not implemented and is reserved.

Syntax

Example Code

Comments

systemFileSelector

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

answer file command, answer folder command, ask file command, dontUseNS property

Summary

Specifies whether the ask file, answer file, and answer folder commands use the standard file dialog boxes built in to the current operating system, or a built-in file dialog box.

Syntax

set the systemFileSelector to {true | false}

Example Code

```
set the systemFileSelector to false
```

Comments

Use the systemFileSelector property to control the appearance of standard file dialogs on Mac OS, OS X, or Windows systems, or to make a cross-platform application use the same file dialog boxes on all platforms instead of using the platform standard.

Value:

The systemFileSelector is true or false.

By default, the systemFileSelector property is set to true.

Comments:

If the systemFileSelector property is set to true, the operating system's own standard file dialogs are used for selecting or naming a file or folder. If the systemFileSelector is false, the application uses its own file dialog boxes.

The setting of this property affects all file dialog boxes used in Revolution, including dialogs displayed by the ask file, answer file, and answer folder commands, as well as dialog boxes displayed by menu items such as `ôSaveö` and `ôOpen Stackö`.

The setting of this property has no effect on Unix systems, where the application's built-in dialog box is always used even if the `systemFileSelector` is set to `true`.

systemVersion

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

buildNumber function, environment function, platform function, QTVersion function, version function, Recipe for finding out whether OS X is running

Summary

Returns the version number of the operating system.

Syntax

the systemVersion
systemVersion()

Example Code

```
the systemVersion  
if the systemVersion is "8.6.0" then set the OSVersion to "8.6"
```

Comments

Use the systemVersion function to determine whether the current version of the operating system supports a particular feature.

Value:

The systemVersion function returns a string.

Comments:

The exact format of the systemVersion varies, depending on the operating system.

On Mac OS and OS X systems, the systemVersion returns three integers separated by decimal points. For example, on Mac OS 8.6, the systemVersion returns 8.6.0.

On Windows systems, the systemVersion returns the internal Windows version designation. The internal designations for several Windows versions are as follows:

Win version	systemVersion function returns:
Windows 95	Windows 4.0

Windows 98	Windows 4.10 (may return other numbers)
Windows Me	Windows 4.90 (may return other numbers)
Windows NT 4	NT 4.0
Windows 2000	NT 5.0
Windows XP	NT 5.1

Because certain sub-versions of Windows 98 and Windows Me return slightly different values for the `systemVersion`, you can check for these versions by testing whether the number is in a range, as in the following example:

```
get word 2 of the systemVersion  
if it >= 4.1 and it < 4.9 then answer "Windows 98!"
```

Note: When running a Mac OS application in the Classic box on an OS X system, the `systemVersion` function returns the version number of the Classic system folder.

systemWindow

property

Synonyms

Objects

stack

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

decorations property, draggable property, maximize keyword, minimize keyword

Summary

Makes a window float above all running applications.

Syntax

set the systemWindow of stack to {true | false}

Example Code

```
set the systemWindow of stack "Magnifier" to true
set the systemWindow of the target to false
```

Comments

Use the systemWindow property to create a stack that can be used as a systemwide utility, available in all applications.

Value:

The systemWindow property of a stack is true or false. By default, the systemWindow of a newly created stack is set to false.

Comments:

The systemWindow property determines the windowÆs layer. If the systemWindow is true, the window floats above all running applications, no matter which application is active.

A stack whose systemWindow property is true has the same appearance as a palette.

On Mac OS systems, the systemWindow property has no effect.

Note: The setting of this property determines whether the stackÆs decorations property includes ôsystemö.

tab

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command, space constant

Summary

Equivalent to the tab character (ASCII 9, Control-I).

Syntax

Example Code

```
write field nextField & tab to file exportFile
```

Comments

Use the tab constant as an easier-to-read substitute for numToChar(9).

Comments:

The tab constant is needed because you can't type the character it represents in a script.

tabbed

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cascade keyword, comboBox keyword, menuMode property, option keyword, popup keyword, pulldown keyword, style property, How to respond to the user clicking a tab in a tabbed window, How to switch between button tabs, Object menu > New Control > Tabbed Button

Summary

Specifies one of the menu types that can be used with the menuMode property.

Syntax

Example Code

```
set the menuMode of button "Dialog Tabs" to tabbed
```

Comments

Use the tabbed keyword to create a set of tabs in a window.

Comments:

Tabbed buttons are normally used to switch between sets of options.

tabGroupBehavior

property

Synonyms

Objects

group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

layer property, radioBehavior property, tabKey message, traversalOn property

Summary

Specifies whether the arrow keys navigate within a group and whether the Tab key skips to the next group.

Syntax

set the tabGroupBehavior of group to {true | false}

Example Code

```
set the tabGroupBehavior of group "Navigation" to true
```

Comments

Use the tabGroupBehavior property to enable using the arrow keys to tab through a group's controls.

Value:

The tabGroupBehavior of a group is true or false.

By default, the tabGroupBehavior property of a newly created group is set to false.

Comments:

If a group's tabGroupBehavior property is set to true, pressing the Tab key moves to the first unlocked field in the group, and pressing it again skips the rest of the controls in the group and moves the selection out of the group.

If the tabGroupBehavior is false, pressing the Tab key moves through the fields and other controls in the group in order, without skipping.

If the tabGroupBehavior is true, the user can press the arrow keys to move from control to control within the group.

If a group's `tabGroupBehavior` is true, you cannot press the Tab key to go to the next tab stop (as set in the `tabStops` property) in any of the fields in the group. Pressing the Tab key moves to the next control in the group instead.

Similarly, if a group's `tabGroupBehavior` is true, you cannot press the arrow keys to move around in any of the fields in the group, regardless of the setting of the `textArrows` property. Pressing an arrow key moves from control to control instead.

tabKey

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

autoTab property, enterKey message, keyDown message, returnKey message, tabStops property, Why doesn't the Tab key move to the next field?, Why doesn't the Tab key move to the next tab stop?, Recipe for shifting the selection to the right or left

Summary

Sent when the user presses the Tab key.

Syntax

tabKey

Example Code

```
on tabKey -- go forward in stack on tab, backward on shift-tab
  if the shiftKey is down then go previous card
  else go next card
end tabKey
```

Comments

Handle the tabKey message when you want to perform an action (such as going to the next card) when the user presses the Tab key.

Comments:

The message is sent to the active (focused) control, or to the current card if no control is focused.

tabStops

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

autoTab property, tabKey message, vGrid property, width property, Why doesn't the Tab key move to the next tab stop?

Summary

Specifies the location of tab stops in a field.

Syntax

set the tabStops of field to tabStopList

Example Code

```
set the tabStops of field "Comments" to 20 -- tab every 20 pixels
set the tabStops of field 1 to 40,80,120,160,200
if the tabStops of me is empty then answer "Sorry - can't tab."
```

Comments

Use the tabStops property to let users tab to a horizontal location within a field.

Value:

The tabStops of a field is a list of one or more positive integers, separated by commas.

By default, the tabStops property of newly created fields is set to empty.

Comments:

If the user presses the Tab key while editing a field, normally the insertion point moves to the next control whose traversalOn property is true. If the tabStops property is true, the user can enter tab characters in the field. Each time the Tab key is pressed, the insertion point moves to the next tab stop on the current line.

The tabStops consists of one or more integers separated by commas. Each integer is the distance in pixels from the left margin of the field to a tab stop. If a tab stop is less than the previous tab stop, the distance is measured relative to the previous tab stop. For example, if the tabStops is set to 20,100,300, tab stops are placed at 20, 100, and 130 pixels from the left margin.

Tip: If you set the field's `vGrid` property to true, a vertical line is drawn at each tab stop. Temporarily setting this property to true can help you get a better idea of where each tab stop is.

If the `tabStops` does not define tabs for the entire width of the field, or if the field's `dontWrap` property is true, Revolution creates implicit tab stops across the entire field. (For example, if a field's `tabStops` is `10`, the field has a tab stop every 10 pixels.) If the `tabStops` property defines more than one tab stop, the width of the last tab column repeats for the width of the field. (For example, if a field's `tabStops` is `20,100,130`, the width of the last tab column is 30, so additional tab stops are automatically created at 160, 190, 220, and so on.)

If the `tabStops` is empty, the field has no tab stops, and pressing the Tab key moves the insertion point to the next field.

Important! Setting the field's `textAlign` to `center` or `right` may cause unexpected results when using tab stops.

Note: If the `tabGroupBehavior` property of a group containing the field is set to true, you cannot press the Tab key to go to the next tab stop. (The `tabStops` setting still controls the spacing of any tab characters that are pasted into the field or placed in it by a handler.)

tan

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

atan function, atan2 function, cos function, pi constant, sin function

Summary

Returns the tangent of an angle (in radians).

Syntax

the tan of angleInRadians
`tan(angleInRadians)`

Example Code

```
tan(pi) -- returns zero  
the tan of myAngle
```

Comments

Use the tan function to find the tangent of an angle.

Parameters:

The angle is a positive or negative number, or an expression that evaluates to a number.

Value:

The tan function returns a number.

Comments:

The tan function repeats every 2π radians: $\tan(x) = \tan(x + 2 * \pi)$, for any value of x .

If the angleInRadians is an odd multiple of $\pi/2$, the tangent is infinite (undefined). The tan function instead returns a very large number for such angles, due to internal number handling.

The tan function requires its input to be in radians. To provide an angle in degrees, use the following custom function:

```
function tanInDegrees angleInDegrees
```

```
    return tan(angleInDegrees * pi / 180)
end cosInDegrees
```

target function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

dragSource function, dragDestination function, me keyword, selectedObject function, target keyword,
How to determine what kind of object triggered a message

Summary

Returns the object which received the message that started execution.

Syntax

the target
target()

Example Code

```
the target  
if word 1 of the target is "button" then clickedAButton  
set the backgroundColor of the target to "black"
```

Comments

Use the target function within a message handler to determine which object originally received the message.

Value:

The target function returns the name property of the object.

Comments:

Suppose a card script contains a mouseDown handler. If the user clicks a button, a mouseDown message is sent to the button. If the button's script does not contain a mouseDown handler, the message is passed to the card, and handled by the card's mouseDown handler.

The target function is similar to the me keyword. In the example described above, within the card's handler, the target function returns the button's name, because the button is the object that first received the mouseDown message. However, the me keyword is a reference to the card, because the card is the object whose script contains the mouseDown handler that is executing.

target
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

me keyword, target function, text property

Summary

Specifies the text contents of the field which was the target of the message currently being handled.

Syntax

Example Code

```
answer target
```

Comments

Use the target keyword to access the contents of the field named in the target function.

Comments:

The only semantic difference between the target function and the target keyword is that the target keyword is not preceded by the word `ötheö`.

If the target object is not a field, the target keyword reports the name of the object, just like the target function.

templateAudioClip

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

audioClip object, play command, reset command, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoClip keyword, How to change default settings for new objects

Summary

Used with the set command to set the properties of newly created audio clip objects.

Syntax

Example Code

```
set the playLoudness of the templateAudioClip to 6
```

Comments

Use the templateAudioClip keyword to set up default properties to be used for any new audio clips you create.

Comments:

The templateAudioClip can be thought of as a special, abstract object. It is not an actual audio clip, but it has all the properties of an audio clip. The ID property of the templateAudioClip is zero.

You can use the set command to set the properties of the templateAudioClip before creating an audio clip. The new audio clip has the properties of the templateAudioClip, by default. Changing the properties of the templateAudioClip does not affect existing audio clips.

The properties of the templateAudioClip are reset to their defaults when all running handlers finish executing.

You can refer to the templateAudioClip using any of the following forms:

templateAudioClip
the templateAudioClip

`templateAudioClip()`

Note: The `templateAudioClip` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

templateButton

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

button object, reset command, templateAudioClip keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoClip keyword, How to change default settings for new objects

Summary

Used with the set command to set the properties of newly created buttons.

Syntax

Example Code

```
set the height of the templateButton to 20
set the textFont of the templateButton to "Arial"
```

Comments

Use the templateButton keyword to set up default properties to be used for any new buttons you create.

Comments:

The templateButton can be thought of as a special, abstract object. It is not an actual button, but it has all the properties of a button. The ID property of the templateButton is zero.

You can use the set command to set the properties of the templateButton before creating a button. The new button has the properties of the templateButton, by default. This example creates a radio button:

```
on makeRadio
  set the style of the templateButton to "radioButton"
  create button "Red" -- "Red" is a radio button
end makeRadio
```

The properties of the templateButton are reset to their defaults when all running handlers finish executing.

You can refer to the `templateButton` using any of the following forms:

- `templateButton`
- `the templateButton`
- `templateButton()`

Note: The `templateButton` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

templateCard

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

reset command, templateAudioClip keyword, templateButton keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoClip keyword, How to change default settings for new objects, How to create a card, How to create a card template for a stack

Summary

Used with the set command to set the properties of newly created cards.

Syntax

Example Code

```
set the backgroundColor of the templateCard to "yellow"
```

Comments

Use the templateCard keyword to set up default properties to be used for any new cards you create.

Comments:

The templateCard can be thought of as a special, abstract object. It is not an actual card, but it has all the properties of a card. The ID property of the templateCard is zero.

You can use the set command to set the properties of the templateCard before creating a card. The new card has the properties of the templateCard, by default. Changing the properties of the templateCard does not affect existing cards.

The properties of the templateCard are reset to their defaults when all running handlers finish executing.

You can refer to the templateCard using any of the following forms:

templateCard

the templateCard

templateCard()

Note: The `templateCard` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

templateEPS

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

reset command, templateAudioClip keyword, templateButton keyword, templateCard keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoClip keyword

Summary

Used with the set command to set the properties of newly created EPS objects.

Syntax

Example Code

```
set the prolog of the templateEPS to field "Set Prolog"
```

Comments

Use the templateEPS keyword to set up default properties to be used for any new EPS objects you create.

Comments:

The templateEPS can be thought of as a special, abstract object. It is not an actual EPS object, but it has all the properties of an EPS. The ID property of the templateEPS is zero.

You can use the set command to set the properties of the templateEPS before creating an EPS object. The new EPS has the properties of the templateEPS, by default. Changing the properties of the templateEPS does not affect existing EPS objects.

The properties of the templateEPS are reset to their defaults when all running handlers finish executing.

You can refer to the templateEPS using any of the following forms:

templateEPS

the templateEPS

templateEPS()

Note: The `templateEPS` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

templateField

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

reset command, templateAudioClip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoClip keyword, How to change default settings for new objects

Summary

Used with the set command to set the properties of newly created fields.

Syntax

Example Code

```
set the textFont of the templateField to myHeadingFont
```

Comments

Use the templateField keyword to set up default properties to be used for any new fields you create.

Comments:

The templateField can be thought of as a special, abstract object. It is not an actual field, but it has all the properties of a field. The ID property of the templateField is zero.

You can use the set command to set the properties of the templateField before creating a field. The new field has the properties of the templateField, by default. Changing the properties of the templateField does not affect existing fields.

The properties of the templateField are reset to their defaults when all running handlers finish executing.

You can refer to the templateField using any of the following forms:

```
templateField  
the templateField  
templateField()
```

Note: The `templateField` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

templateGraphic

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

reset command, templateAudioClip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoClip keyword, How to change default settings for new objects

Summary

Used with the set command to set the properties of newly created graphics.

Syntax

Example Code

```
set the style of the templateGraphic to rectangle
```

Comments

Use the templateGraphic keyword to set up default properties to be used for any new graphics you create.

Comments:

The templateGraphic can be thought of as a special, abstract object. It is not an actual graphic, but it has all the properties of a graphic. The ID property of the templateGraphic is zero.

You can use the set command to set the properties of the templateGraphic before creating a graphic. The new graphic has the properties of the templateGraphic, by default. Changing the properties of the templateGraphic does not affect existing graphics.

The properties of the templateGraphic are reset to their defaults when all running handlers finish executing.

You can refer to the templateGraphic using any of the following forms:

templateGraphic

the templateGraphic

`templateGraphic()`

Note: The `templateGraphic` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

templateGroup

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

reset command, templateAudioClip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoClip keyword, How to change default settings for new objects

Summary

Used with the set command to set the properties of newly created groups.

Syntax

Example Code

```
set the lockLoc of the templateGroup to true
```

Comments

Use the templateGroup keyword to set up default properties to be used for any new groups you create.

Comments:

The templateGroup can be thought of as a special, abstract object. It is not an actual group, but it has all the properties of a group. The ID property of the templateGroup is zero.

You can use the set command to set the properties of the templateGroup before creating a group. The new group has the properties of the templateGroup, by default. Changing the properties of the templateGroup does not affect existing groups.

The properties of the templateGroup are reset to their defaults when all running handlers finish executing.

You can refer to the templateGroup using any of the following forms:

- templateGroup
- the templateGroup
- templateGroup()

Note: The `templateGroup` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

templateImage

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

reset command, templateAudioClip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoClip keyword, How to change default settings for new objects

Summary

Used with the set command to set the properties of newly created images.

Syntax

Example Code

```
set the dontDither of the templateImage to true
```

Comments

Use the templateImage keyword to set up default properties to be used for any new images objects you create.

Comments:

The templateImage can be thought of as a special, abstract object. It is not an actual image, but it has all the properties of an image. The ID property of the templateImage is zero.

You can use the set command to set the properties of the templateImage before creating an image. The new image has the properties of the templateImage, by default. Changing the properties of the templateImage does not affect existing images.

The properties of the templateImage are reset to their defaults when all running handlers finish executing.

You can refer to the templateImage using any of the following forms:

templateImage

the templateImage

`templateImage()`

Note: The `templateImage` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

templatePlayer

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

reset command, templateAudioClip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templateScrollbar keyword, templateStack keyword, templateVideoClip keyword, How to change default settings for new objects

Summary

Used with the set command to set the properties of newly created players.

Syntax

Example Code

```
set the alwaysBuffer of the templatePlayer to true
```

Comments

Use the templatePlayer keyword to set up default properties to be used for any new players you create.

Comments:

The templatePlayer can be thought of as a special, abstract object. It is not an actual player, but it has all the properties of a player. The ID property of the templatePlayer is zero.

You can use the set command to set the properties of the templatePlayer before creating a player. The new player has the properties of the templatePlayer, by default. Changing the properties of the templatePlayer does not affect existing players.

The properties of the templatePlayer are reset to their defaults when all running handlers finish executing.

You can refer to the templatePlayer using any of the following forms:

templatePlayer

the templatePlayer

templatePlayer()

Note: The `templatePlayer` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

templateScrollbar

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

reset command, templateAudioClip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateStack keyword, templateVideoClip keyword, How to change default settings for new objects

Summary

Used with the set command to set the properties of newly created scrollbars.

Syntax

Example Code

```
set the style of the templateScrollbar to scale
```

Comments

Use the templateScrollbar keyword to set up default properties to be used for any new scrollbars you create.

Comments:

The templateScrollbar can be thought of as a special, abstract object. It is not an actual scrollbar, but it has all the properties of a scrollbar. The ID property of the templateScrollbar is zero.

You can use the set command to set the properties of the templateScrollbar before creating a scrollbar. The new scrollbar has the properties of the templateScrollbar, by default. Changing the properties of the templateScrollbar does not affect existing scrollbars.

The properties of the templateScrollbar are reset to their defaults when all running handlers finish executing.

You can refer to the templateScrollbar using any of the following forms:

templateScrollbar

the templateScrollbar

`templateScrollbar()`

Note: The `templateScrollbar` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

templateStack

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

reset command, stack object, templateAudioClip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateVideoClip keyword, How to change default settings for new objects

Summary

Used with the set command to set the properties of newly created stacks.

Syntax

Example Code

```
set the ID of the templateStack to 2000
```

Comments

Use the templateStack keyword to set up default properties to be used for any new stacks you create.

Comments:

The templateStack can be thought of as a special, abstract object. It is not an actual stack, but it has all the properties of a stack. The ID property of the templateStack is 1001.

You can use the set command to set the properties of the templateStack before creating a stack. The new stack has the properties of the templateStack, by default. Changing the properties of the templateStack does not affect existing stacks.

The properties of the templateStack are reset to their defaults when all running handlers finish executing.

You can refer to the templateStack using any of the following forms:

templateStack

the templateStack

templateStack()

Note: The `templateStack` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

templateVideoClip

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

reset command, templateAudioClip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, How to change default settings for new objects

Summary

Used with the set command to set the properties of newly created video clips.

Syntax

Example Code

```
set the scale of the templateVideoClip to 2.0 -- double size
```

Comments

Use the templateVideoClip keyword to set up default properties to be used for any new video clips you create.

Comments:

The templateVideoClip can be thought of as a special, abstract object. It is not a actual video clip, but it has all the properties of a video clip. The ID property of the templateVideoClip is zero.

You can use the set command to set the properties of the templateVideoClip before creating a video clip. The new video clip has the properties of the templateVideoClip, by default. Changing the properties of the templateVideoClip does not affect existing video clips.

The properties of the templateVideoClip are reset to their defaults when all running handlers finish executing.

You can refer to the templateVideoClip using any of the following forms:

templateVideoClip

the templateVideoClip

`templateVideoClip()`

Note: The `templateVideoClip` keyword is implemented internally as a function and appears in the `functionNames`. However, it cannot be used as a function in an expression.

tempName

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

create folder command, filename property, open file command, specialFolderPath function

Summary

Returns a unique file name in the folder the operating system uses for temporary files.

Syntax

the tempName
tempName()

Example Code

```
the tempName  
put the tempName into testDataPath
```

Comments

Use the tempName function to find an appropriate place to put a temporary file.

Value:

The tempName function returns an absolute file path.

Comments:

You can use temporary files to store intermediate data, downloaded URLs, and other material that is more conveniently placed in a file but does not need to be saved permanently.

Using the tempName function does not create the file, only gets a suggested file name. Use the open file command to create the file.

The file name returned by the tempName function is one that doesn't exist, so you can use it safely. If you need to create another file, use the tempName function again to obtain a new file name.

The location of the temporary files depends on the operating system. Each operating system provides its own location for temporary files.

ten

constant

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command, tenth keyword

Summary

Equivalent to the number 10.

Syntax

Example Code

```
if ten is in field "Score" then beep
```

Comments

Use the ten constant when it is easier to read than the number 10.

tenth

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

last keyword, middle keyword, number property, ten constant

Summary

Designates the tenth member of a set.

Syntax

Example Code

```
put the ID of tenth card into myID  
multiply the tenth word of currentAmounts by ten
```

Comments

Use the tenth keyword in an object reference or chunk expression.

Comments:

The tenth keyword can be used to specify any object whose number property is 10. It can also be used to designate the tenth chunk in a chunk expression.

The word the is optional when using the tenth keyword.

text

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

dateTime keyword, numeric keyword, sort command, sort container command, text property

Summary

Used with the sort and sort container commands to sort normally, character by character.

Syntax

Example Code

```
sort cards descending text by field "State"  
sort lines of the target text by word 2 of each
```

Comments

Use the text keyword to improve the clarity of your code. For example, if a handler contains several sort commands and some are numeric, you can use the text keyword explicitly to point up the fact that there are both alphabetic and numeric sorts.

Comments:

Since text is the default sort order, you never actually need to use the text keyword; if you leave it out, the sort is performed alphabetically anyway.

text property

Synonyms
textData

Objects
button, field, image

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
label property, me keyword, menuHistory property, name property, target function, target keyword,
How to display the contents of a text file, How to display the same text on multiple cards, How to import
a text file into a field, How to respond to a change in field contents

Summary
Specifies the text contained by a button or field or the binary data in an image.

Syntax
set the text of {button | field | image} to string

Example Code
put the text of button "Popup" into menuContents
set the text of field "Display Script" to the script of the target

Comments
Use the text property to work with the contents of a field, button, or image.

Value:
The text of a field, button, or image is a string.

Comments:
The text of a field is simply the contents of the field. You can also use the target keyword to obtain the
contents of a field:

```
the text of the target is target
-- evaluates to true if target is a field
```

The text of a button is used as the button's menu contents. (A menu whose contents is the text of a
button is called a button menu.) If the button's style is not set to "menu", the setting of its text
property has no effect. However, the text property can be used to store data in a button, regardless of the
button's style; the text is not visible to the user unless the button is a button menu.

The text of an image is the binary data that makes up the image.

Note: Indeed, it is odd to be calling binary data the `text` property. The usage came about for historical reasons.

These two statements are equivalent:

```
set the text of object to string  
put string into object
```

Tip: Instead of displaying a button's text when using the button as a menu, you can designate a stack to be displayed as the menu contents. This type of menu is called a stack menu. If the button's `menuName` property is not empty, the stack specified by the `menuName` is used as the menu contents, instead of the button's text property.

textAlign

property

Synonyms

Objects

button, field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

dontWrap property, foregroundColor property, margins property, printTextAlign property, textFont property, textSize property, textStyle property, How to change the alignment of text, Text menu > Align

Summary

Specifies how text is aligned or justified in a field or button.

Syntax

set the textAlign of {button | field} to {left | center | right}

Example Code

```
set the textAlign of field "Results Notification" to center
```

Comments

Use the textAlign property to change the appearance of fields or buttons.

Value:

The textAlign is `left`, `center`, or `right`.

By default, the textAlign of newly created fields and buttons is `left`.

Comments:

Unlike other text properties, the setting of the textAlign property is not inherited from the field's or button's owner. Setting the textAlign of a field or button to empty results in the property being set to `left`.

textArrows

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

arrowKey message, go command, navigationArrows property, How to block or change the action of arrow keys, Shortcut to go to the top or bottom of a field, Shortcut to navigate in a field

Summary

Specifies whether the arrow keys move the insertion point when there is a text selection.

Syntax

set the textArrows to {true | false}

Example Code

```
set the textArrows to true
```

Comments

Use the textArrows property to control the application's response to arrow keys.

Value:

The textArrows is true or false.

By default, the textArrows property is set to true.

Comments:

If there is an insertion point or text selection in a field, and the textArrows is true, the arrow keys move the insertion point one character left or right, or one line up or down.

If there is no insertion point or text selection, the setting of the textArrows property has no effect.

If a group's tabGroupBehavior is true, you cannot press the arrow keys to move around in any of the fields in the group, regardless of the setting of the textArrows property. Pressing an arrow key moves from control to control instead.

textFont

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

charSet property, fontNames function, printTextFont property, scriptTextFont property, textSize property, textStyle property, How to enter or display Unicode text in a field, How to store styled text in a variable or property, Text menu > Font, Shortcut to remove font changes from text

Summary

Specifies the font face of text in an object.

Syntax

set the textFont of object to fontName[,language]

set the textFont of [chunk of] field to fontName[,language]

Example Code

```
set the textFont of button "Hello" to "Courier"
set the textFont of the foundChunk to "Comic Sans"
set the textFont of field "Input" to "Arial,Japanese"
put item 2 of the textFont of the selectedChunk into theLanguage
set the textFont of field "Text" to ",Japanese"
```

Comments

Use the textFont property to change the appearance of text.

Value:

The textFont of an object or chunk is either empty or a string.

If the textFont includes a second item, it is one of the following:

ò ANSI (synonym for ôEnglishö)

ò Arabic

ò Bulgarian

ò Chinese

ò English

ò Greek

ò Hebrew

ò Japanese (Shift-JIS)

- ò Korean
- ò Roman
- ò Russian (Cyrillic)
- ò Thai
- ò Turkish
- ò SimpleChinese (Simplified Chinese)
- ò Ukrainian
- ò Unicode (UTF-16)
- ò w (synonym for ôUnicodeö)

Comments:

Setting the textFont of an object to empty allows the textFont of the objectÆs owner to show through. Use the effective keyword to find out what font is used for the object, even if its own textFont is empty.

If the chunk contains more than one font, the chunkÆs textFont property reports ômixedö.

Note: Setting the textFont to empty also sets the textSize and textStyle properties of the object or chunk to empty.

To specify a Unicode font, append ôUnicodeö or the desired language name to the font name, separating it with a comma. For example, to set the font face of a button to a Japanese font, use a statement like the following:

```
set the textFont of button 1 to "Osaka,Japanese"
-- displays button label in Japanese, in the Osaka font
```

If the textFont of a field or a chunk includes a language, selecting in that field or chunk automatically switches the keyboard input method to the one for that language, ready to type text in the language. The htmlText property of the field or chunk uses the language to derive the HTML ôlangö attribute.

On Mac OS systems, if the specified font isnÆt available, the system font (which is set in the Appearance control panel and specifies the font used for menus) is used. On Unix systems, if the specified font isnÆt available, Helvetica is used. On Windows systems, if the specified font isnÆt available, the current Windows font is used.

If the specified Unicode font is unavailable but the system supports Unicode, Revolution uses the default font for the specified language. If you specify a language, but leave the fontName empty, the object uses the default font for the specified language, as specified by the operating systemÆs settings.

Changes to Transcript:

The ability to specify ôUnicodeö or a language name was introduced in version 2.0. In previous versions, display of double-byte characters was not supported.

textHeight

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

effective keyword, fixedLineHeight property, formattedHeight property, printTextHeight property, textSize property

Summary

Specifies the amount of space between lines of text in a field.

Syntax

set the textHeight of field to pixels

Example Code

```
get the effective textHeight of field "Results"  
set the textHeight of field 1 to theLines * the textHeight of field 1
```

Comments

Use the textHeight property to change the appearance of text in a field.

Value:

The textHeight of a field is an integer greater than or equal to 4.

By default, the textHeight property of newly created fields is set to 4/3 times the textSize of the field.

Comments:

The textHeight property determines how tall each line of a field is, and is equal to the vertical distance in pixels between the baseline of one line and the baseline of the next. The minimum textHeight is 4 pixels.

If you change the field's textSize, Revolution automatically sets the textHeight to trunc(4/3 * the textSize of field). For example, if the textSize is set to 12, the textHeight is set to 16 by default. If the textSize is set to 14, the textHeight is set to 18.

If a field's fixedLineHeight property is false, the textHeight of the field has no effect: in this case, the text height of each line is determined by the size of the largest text in the line, and changing the textHeight doesn't change the spacing of lines.

textHeightSum

function

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

formattedHeight property, formattedWidth property, pageHeights property, textHeight property

Summary

Returns the total height of the text in a field or button.

Syntax

the textHeightSum of object

textHeightSum(object)

Example Code

```
the textHeightSum of field "Comments"
```

```
if the textHeightSum of group 1 > the height of this card then expandCd
```

Comments

Use the textHeightSum function to determine how much vertical space a fieldÆs or buttonÆs text requires.

Parameters:

The object is a field, button, card, or group.

Value:

The textHeightSum returns a positive integer.

Comments:

This function exists for compatibility with imported SuperCard projects. You can use the formattedHeight property to find the height of the text in a field or button.

If the object is a button or field, the textHeightSum returns the number of vertical pixels needed to display all the text at the fieldÆs current width. If the object is a card or group, the textSumHeight returns the number of vertical pixels needed to display all the objects in the card or group.

textShift

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

fixedLineHeight property, textHeight property, textSize property, How to make subscripts and superscripts, Text menu > Subscript, Text menu > Superscript

Summary

Specifies how far text is shifted up or down from its baseline.

Syntax

set the textShift of [chunk of] field to pixels

Example Code

```
set the textShift of last char of word 2 of field 1 to 3 -- subscript  
set the textShift of char 12 of the selectedField to -1 -- superscript
```

Comments

Use the textShift property to subscript and superscript chunks of text in a field.

Value:

The textShift of a field is an integer.

By default, the textShift property of newly created fields is set to zero.

Comments:

The textShift is the number of pixels to shift the text down from the baseline, creating a subscript. If the textShift is negative, the text is shifted up (superscripted). If the textShift is zero, the text is placed in normal position, not shifted up or down.

Note: If the field's fixedLineHeight property is true, you may need to increase the field's textHeight to make room for subscripts and superscripts.

textSize

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

fixedLineHeight property, fontSizes function, printTextSize property, scriptTextSize property, textFont property, textHeight property, textShift property, textStyle property, How to store styled text in a variable or property, Text menu > Size, Shortcut to remove font changes from text

Summary

Specifies the point size of text displayed by an object.

Syntax

set the textSize of object to pointSize

set the textSize of [chunk of] field to pointSize

Example Code

```
set the textSize of card "Intro" to 24
set the textSize of word 1 of field ID 3 to 9
```

Comments

Use the textSize property to change the appearance of text.

Value:

The textSize of an object or chunk is either empty or an integer greater than 4.

Comments:

Setting the textSize of an object to empty allows the textSize of the object's owner to show through.

Use the effective keyword to find out what size is used for the object, even if its own textSize is empty.

textStyle

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bold keyword, borderColor property, bottomColor property, box keyword, condensed keyword, expanded keyword, fontStyles function, italic keyword, link keyword, printTextStyle property, strikeouts keyword, textFont property, textShift property, textSize property, threeDBox keyword, topColor property, underline keyword, underlineLinks property, How to remove a single style from text, How to remove all styles from text, How to store styled text in a variable or property, Why is some text blue and underlined?, Shortcut to remove font changes from text

Summary

Specifies the style or styles applied to text in an object.

Syntax

set the textStyle of object to {empty | plain | stylesList}

set the textStyle of [chunk of] field to {empty | plain | stylesList}

Example Code

```
set the textStyle of line 2 of field "Choices" to "italic,underline"
set the textStyle of the selectedChunk to "box"
set the textStyle of button "Off" to "italic,bold"
```

Comments

Use the textStyle property to change the appearance of text.

Value:

The textStyle of an object or chunk is either `plain`, `empty`, `mixed`, or one or more of the following, separated by commas:

- ò bold
- ò italic
- ò underline
- ò strikeouts
- ò box
- ò threeDbox
- ò link (or group)

ò condensed
ò expanded

Comments:

Setting the textStyle to ôplainö turns off all styles. (Setting the textStyle to ôplainö plus one or more additional styles may result in anomalous behavior.)

Setting the textStyle of an object to empty allows the textStyle of the objectÆs owner to show through. Use the effective keyword to find out what style is used for the object, even if its own textStyle is empty. Similarly, use the effective keyword to find out what style is used for a chunk of text, even if the chunkÆs textStyle is empty.

If you request the textStyle of a chunk which includes multiple style runs—for example, some bold and some plain text—the property reports ômixedö.

The ôlinkö style can be used only for chunks of a field, not for objects. Setting the textStyle of a chunk to ôlinkö turns it into a text group.

Changes to Transcript:

The ôlinkö style was introduced in version 1.1. In previous versions, the term ôgroupö was used. The term ôgroupö is still available as a partial synonym: setting the textStyle of a chunk to ôgroupö sets it to ôlinkö.

the
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

of keyword, this keyword

Summary

Appears before the names of properties and the names of functions that are not followed by parentheses

Syntax

Example Code

```
set the script of me to field "Script Contents"  
put the date into field "Current Date"
```

Comments

Use the the keyword to unambiguously indicate how an expression should be parsed.

Comments:

The the keyword is mandatory before the names of built-in functions, unless the parentheses form of the function is used:

```
get the time -- must use "the" if () are not used  
get time()  -- cannot use "the" if () are used
```

The the keyword is optional before the names of properties, but it is good style to include it.

then

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

else keyword, end if keyword, if control structure

Summary

Used in an if control structure to separate the condition from the statements that are executed if the condition is true.

Syntax

Example Code

```
if a is true then beep
```

Comments

Use the then keyword before the statements you want to execute if a condition is true.

Comments:

If you want to execute a single statement, place the then keyword before the statement on the same line, as in these examples:

```
if 1+1 = 2 then doSomethingTrite
```

```
if someConditionObtains()  
then performAnAction
```

If you want to execute multiple statements, place the then keyword on the line before the list of statements, as in this example:

```
if the backgroundColor of this stack is white then  
  doFirstAction  
  doAnotherAction  
end if
```


there is a
operator

Synonyms
there is an

Objects
logical

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
cardNames property, exists function, files function, folders function, groupNames property, openProcesses function, openProcessIDs function, there is no operator, volumes function, Operator Precedence Reference

Summary
Evaluates to true if the specified object, file, folder, or process exists, false otherwise.

Syntax
there is a {object | file filePath | folder folderPath | process procName}

Example Code
there is a card button 1 -- evaluates to true if any card buttons exist
if there is a card ID 3445 then go card ID 3445
put (there is a file it) into fileIsThere

Comments
Use the there is a operator to make sure an object or file exists before working with it, or to find out whether there are any objects of a specified type.

Parameters:
The object is any object reference.

The filePath specifies the name and location of a file you want to check. If you specify a name but not a location, Revolution looks for the file in the defaultFolder.

The folderPath specifies the name and location of a folder you want to check. If you specify a name but not a location, Revolution looks for the folder in the defaultFolder.

The procName is the name of any process that Revolution started.

Comments:

The `there is a` operator is the logical inverse of the `there is no` operator. When one is true, the other is false.

The expression `there is a stack stackName` evaluates to true if the stack is open, if it is closed but loaded into memory, or if it appears in the `stackFiles` property of any open stack.

Tip: To find out whether a stack is open, check its `mode` property. A stack that is loaded into memory but is not open has a mode of zero.

Note: The `there is a` operator is implemented internally as a function, and therefore has lower precedence than other operators.

there is no
operator

Synonyms

there is not a, there is not an

Objects

logical

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

exists function, is not a operator, not operator, there is a operator, Operator Precedence Reference

Summary

Evaluates to false if the specified object, file, or folder, or process exists, true otherwise.

Syntax

there is no {object | file filePath | folder folderPath | process procName}

Example Code

```
there is no group "Cranky"  
if there is no player 1 then create player  
put (there is not a folder "Temp") into safeToCreate
```

Comments

Use the there is no operator to make sure an object or file does not yet exist before creating it, or that a program is not already running before launching it.

Parameters:

The object is any object reference.

The filePath specifies the name and location of a file you want to check. If you specify a name but not a location, Revolution looks for the file in the defaultFolder.

The folderPath specifies the name and location of a folder you want to check. If you specify a name but not a location, Revolution looks for the folder in the defaultFolder.

The procName is the name of any process that Revolution started.

Comments:

The there is no operator is the logical inverse of the there is a operator. When one is true, the other is false.

Note: The there is no operator is implemented internally as a function, and therefore has lower precedence than other operators.

third

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

last keyword, middle keyword, number property, three constant

Summary

Designates the third member of a set.

Syntax

Example Code

```
set the hilite of the third button to true  
repeat until third line of keysPressed is "Control"
```

Comments

Use the third keyword in an object reference or chunk expression.

Comments:

The third keyword can be used to specify any object whose number property is 3. It can also be used to designate the third chunk in a chunk expression.

The word the is optional when using the third keyword.

this

keyword

Synonyms

current

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

the keyword, How to find out which card is the current card

Summary

Indicates the current stack, or the current card of the current stack.

Syntax

Example Code

```
put the name of this card into currentName
sort cards by the mark of this card
send preOpenStack to this stack
```

Comments

Use the this keyword to indicate the current card or current stack in an expression.

Comments:

When used in a sort command, the this keyword indicates each card to be sorted. You can use the this keyword to sort cards by a card property.

three
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, third keyword

Summary

Equivalent to the number 3.

Syntax

Example Code

```
repeat three times -- same as "repeat 3 times"
```

Comments

Use the three constant when it is easier to read than the numeral 3.

threeD

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

borderWidth property, bottomColor property, bottomPattern property, shadow property, showBorder property, threeDHilite property, topColor property, topPattern property

Summary

Specifies whether an object appears to stick out of or recede into the screen.

Syntax

set the threeD of object to {true | false}

Example Code

```
set the threeD of last button to true
```

Comments

Use the threeD property to determine whether an object appears flat or three-dimensional.

Value:

An object's threeD property is true or false.

By default, the threeD property of newly created objects is true.

Comments:

If the threeD of an object is true, the object appears to stick up out of the screen (if it is a button) or be pressed a few pixels below the level of the screen. If the threeD is false, the object appears flat.

The borders of objects whose threeD is true are drawn with the topColor and bottomColor to simulate depth. The borders of objects whose threeD is false are drawn with the borderColor.

If the object's borderWidth property is zero, or its showBorder is false, the threeD property has no effect.

You can set the threeD property of a stack, but the property has no effect.

threeDBox

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomColor property, box keyword, textStyle property, topColor property, Text menu > 3D Box

Summary

Used with the textStyle property to specify that a chunk of text in a field has a three-dimensional box drawn around it.

Syntax

Example Code

```
set the textStyle of line 1 to 3 of field "Table" to threeDBox
```

Comments

Use the threeDBox keyword to emphasize text by drawing a box around it.

Comments:

The top and left edges of the box are drawn in the object's bottomColor and bottomPattern. The bottom and right edges of the box are drawn in the object's topColor and topPattern.

If the text crosses more than one screen line, a box is drawn around each line of the text. If the text has mixed styles, a separate box is drawn around each style run. For example, if three words are boxed and the last is also boldfaced, a box is drawn around the first two words, and another box is drawn around the boldfaced word.

An object or chunk of a field may have the style box or threeDBox, but not both at once.

threeDHilite

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomColor property, bottomPattern property, effective keyword, hilitedLine property, listBehavior property, noncontiguousHilites property, selectedText function, threeD property, toggleHilites property, topColor property, topPattern property

Summary

Specifies whether selected lines in a list field look like they're receding into the screen.

Syntax

set the threeDHilite of field to {true | false}

Example Code

```
set the threeDHilite of field "Options" to false
set the threeDHilite of the fieldTemplate to true
```

Comments

Use the threeDHilite property to control the appearance of list fields.

Value:

The threeDHilite of a field is true or false.

By default, the threeDHilite property of newly created fields is set to false.

Comments:

If the threeDHilite property is true, selected lines in a list field are drawn with a three-D box around them. (The box is drawn using the field's effective topColor and bottomColor properties.) If the threeDHilite is false, the selected lines are simply highlighted.

If the field's listBehavior property is set to false, this property has no effect.

throw

control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

catch keyword, return control structure, try control structure

Summary

Returns an error message to a try control structure.

Syntax

throw errorString

Example Code

Comments

Use the throw control structure in a handler called from within a try control structure.

Form:

The throw statement appears on a line by itself, anywhere inside a handler.

Parameters:

The errorString is the string that is returned to the calling try control structure. The errorString becomes the parameter of the catch lin in the try control structure.

Comments:

If Revolution generates the error (for example, an execution error from a built-in command), it returns a positive number to the try control structure. To avoid confusion, therefore, a throw control structure should return a negative number, or a non-numeric string.

If a throw control structure is executed in a handler that was not called from within a try control structure, an errorDialog message is sent to the object, and the errorString is passed as a parameter.

Note: The throw control structure is implemented internally as a command and appears in the commandNames.

thumbPosition

property

Synonyms

thumbPos

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

endValue property, showValue property, startValue property, thumbSize property

Summary

Specifies the current position of a scrollbar's draggable thumb.

Syntax

set the thumbPosition of scrollbar to number

Example Code

```
set the thumbPosition of scrollbar 1 to 400
set the thumbPosition of me to (3 * the thumbSize of me)
```

Comments

Use the thumbPosition to move a scrollbar's draggable thumb when the content it controls is scrolled, or to check where the thumb has been dragged in order to scroll the content.

Value:

The thumbPosition of a scrollbar is a number between the scrollbar's startValue and endValue.

By default, the thumbPosition property of newly created scrollbars is set to zero (the default startValue).

Comments:

The thumbPosition is the location of the top edge (for vertical scrollbars) or left edge (for horizontal scrollbars) of the scrollbar thumb, using the scale set by the scrollbar's startPosition and endPosition properties.

If the style of the scrollbar is scale, the maximum thumbPosition, when the scrollbar thumb is all the way to the bottom (for a vertical scrollbar) or right (for a horizontal scrollbar), is the endValue.

If the style of the scrollbar is scrollbar or progress, the maximum value of the thumbPosition is the scrollbar's endValue minus the thumbSize.

thumbSize

property

Synonyms

Objects

scrollbar

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

endValue property, formattedHeight property, formattedWidth property, proportionalThumbs property, startValue property, thumbPosition property

Summary

Specifies how large a scrollbarÆs draggable thumb is.

Syntax

set the thumbSize of scrollbar to number

Example Code

```
set the thumbSize of scrollbar "Contents Group" to 500
set the thumbSize of scrollbar 1 to
```

```
(the height of group "Main"/the formattedHeight of group "Main")
```

Comments

Use the thumbSize property to specify the size of a scrollbarÆs draggable thumb.

Value:

The number is a number between the scrollbarÆs startValue and endValue.

Comments:

Normally, the thumbSize is proportional to the proportion of content visible on the screen. For example, if a third of the content can be seen and the rest is always scrolled off, the scrollbar thumb should cover a third of the scrollbar area, so the number should be one-third of the difference between the scrollbarÆs startValue and its endValue.

If the proportionalThumbs property is set to false, this property has no effect.

ticks

function

Synonyms
tick

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

convert command, date function, milliseconds function, seconds function, ticks keyword, time function

Summary

Returns the number of ticks since the start of the eon.

Syntax

the ticks
ticks()

Example Code

```
the ticks  
put the ticks - startTicks into elapsedTicks
```

Comments

Use the ticks function to time events that must be checked more often than once per second.

Value:

The ticks function returns a positive integer.

Comments:

Unlike HyperTalk's `ticks` function, the ticks function returns the total number of ticks since midnight GMT, January 1, 1970, rather than the total number of ticks since the last system startup.

A tick is one-sixtieth of a second. If you don't need to time events with this much precision, use the seconds function. If you need to time events with greater precision, use the milliseconds function.

ticks

keyword

Synonyms

tick

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

milliseconds keyword, seconds keyword, ticks function

Summary

Designates the number of ticks in a time period.

Syntax

Example Code

```
wait for 10 ticks -- 1/6 of a second
```

Comments

Use the ticks keyword to designate a time period with the wait or send commands.

Comments:

When used with the wait or send commands, the ticks keyword designates a time period measured in sixtieths of a second.

tilt

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

See Also:

constraints property, nodes property, pan property, zoom property

Summary

Specifies the current vertical view angle of a QuickTime VR movie.

Syntax

set the tilt of player to degrees

Example Code

```
set the tilt of player "Arctic" to -30
put (the tilt of me) / 2 into halfwayPoint
```

Comments

Use the tilt property to find out where the user is in a QuickTime VR movie.

Value:

The tilt is a number between -90 and 90.

Comments:

The user can move the view of a QuickTime VR movie using the navigational controls in the player; a handler can change the view by setting the player's pan and tilt properties.

The tilt specifies the amount of rotation in the vertical plane, in degrees. (Think of a person standing in the middle of a scene and tilting his or her head up and down. The point where the person is standing is the `currentNode`, and the amount the person's head is tilted is the tilt.)

A tilt of zero corresponds to the default view of the scene, with the viewer looking straight ahead. As the viewer's head tilts up, the tilt increases: when the tilt is 90, the viewer is looking straight up. As the viewer's head tilts down, the tilt decreases: when the tilt is -90, the viewer is looking straight down.

The tilt is limited by the player's constraints property. If you specify a tilt greater than the range permitted by the constraints, the tilt is set to the highest permitted value. If you specify a tilt less than the range permitted by the constraints, the tilt is set to the lowest permitted value.

If the player does not contain a QuickTime VR movie, its tilt property is zero.

time

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

abbreviated keyword, convert command, date function, english keyword, internet keyword, long keyword, milliseconds function, seconds function, short keyword, system keyword, ticks function, twelveHourTime property, How to delay an action, Recipe for storing a modification timestamp

Summary

Returns the current time.

Syntax

the [long | abbr[rev[iated]] | short] [english | system] time
time()

Example Code

```
put the time into the lastModifiedTime of this card  
set the label of button "Clock" to the short system time
```

Comments

Use the time function to display the current time to the user, or to store the time for later use.

Value:

If the useSystemDate property is set to true or if you specify the system time, the times returned by the time function are formatted according to the user's system preferences.

If the useSystemDate is false or if you specify the english time, the times are in the format described below:

The time form returns the hour and minute separated by a colon, a space, and "AM" or "PM".

The short time form returns the same value as the time form.

The abbreviated time form returns the same value as the time form.

The long time form returns the hour, minute, and second separated by colons, a space, and `AM` or `PM`.

Comments:

If the `twelveHourTime` property is set to `false`, the value returned by the time function does not include `AM` or `PM`.

The format of the system time forms is set by the Date & Time control panel (on Mac OS systems), the Date control panel (on Windows systems), or the `LANG` environment variable (on Unix systems).

Changes to Transcript:

The ability to use the time format preferred by the user was introduced in version 1.1. In previous versions, the time function, along with the date function, consistently used the standard U.S. format, even if the operating system's settings specified another language or time format.

timeScale

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

currentTime property, duration property, endTime property, seconds function, startTime property

Summary

Reports the number of intervals per second of a movie or sound.

Syntax

get the timeScale of player

Example Code

```
put the timeScale of last player into intervalsPerSecond  
put (the duration of me/the timeScale of me) into runTime
```

Comments

Use the timeScale property to convert internal movie or sound times into seconds.

Value:

The timeScale of a player is a positive integer.

This property is read-only and cannot be set.

Comments:

The timeScale is the number of intervals per second of a movie or sound. These intervals are used for the player's startTime, endTime, duration, and currentTime properties, and you can use the timeScale property to convert from the time scale used by the movie or sound to seconds.

title

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

decorations property, maximize keyword, minimize keyword, How to prevent the user from moving a window, How to remove a window's title bar

Summary

Used with the decorations property to indicate whether the window has a title bar.

Syntax

Example Code

```
set the decorations of this stack to "title,minimize"
```

Comments

Use the title keyword to turn a window's title bar on or off.

Comments:

Setting a stack's decorations property to maximize, minimize, or menu automatically includes title in the value of the decorations, turning on the window's title bar.

titleWidth

property

Synonyms

labelWidth

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

label property, menuMode property, name property, width property

Summary

Specifies the amount of a button's width used to display the button's name or label property to the left of the button.

Syntax

set the titleWidth of button to pixels

Example Code

```
set the titleWidth of button "Options" to 90  
set the titleWidth of button "Standard" to 0
```

Comments

Use the titleWidth property to create popup menus with a title.

Value:

The titleWidth of a button is a non-negative integer.

By default, the titleWidth property of newly created buttons is set to zero.

Comments:

The titleWidth is taken from the total width of the button. This means that setting a titleWidth makes the button narrower, and you may have to increase the width property to accommodate both title and button.

to
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

from keyword

Summary

Used with several commands, usually to designate a destination point.

Syntax

Example Code

```
drag image "sprite" from firstPoint to secondPoint with shiftKey  
copy myButton to this card  
seek to 102 in file "/usr/bin/supfiles/man"  
set the isRaining of me to true
```

Comments

Use the to keyword to complete a command that requires it.

Comments:

The to keyword is used with the add, convert, copy, create alias, drag, exit to top, export, go, move, open socket, post, print card, rename, seek, send, send to program, write to file, write to process, write to socket, and write to driver commands.

It is also used to designate the final color in visual effects, and with the set command to designate the new property setting.

toggleHilites

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

hilitedLine property, listBehavior property, lockText property, noncontiguousHilites property, selectedText function

Summary

Specifies whether clicking a selected line in a list field deselects that line.

Syntax

set the toggleHilites of field to {true | false}

Example Code

```
set the toggleHilites of field "Movies" to true
```

Comments

Use the toggleHilites property to control the behavior of list fields.

Value:

The toggleHilites of a field is true or false.

By default, the toggleHilites property of newly created fields is set to false.

Comments:

If a field's toggleHilites property is set to true, clicking a selected line deselects the line. If the toggleHilites is false, clicking a selected line does nothing.

If the field's listBehavior property is false, the setting of the toggleHilites property has no effect.

Important! Setting the toggleHilites to true automatically sets the field's noncontiguousHilites and multipleLines properties to true. (However, setting the toggleHilites to false does not set these properties back to false.) To set the toggleHilites to true and the noncontiguousHilites or multipleLines to false, be sure to set the toggleHilites first, then set the other properties to false.

token

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

character keyword, int1 keyword, int2 keyword, int4 keyword, item keyword, line keyword, real4 keyword, real8 keyword, uInt1 keyword, uInt2 keyword, uInt4 keyword, word keyword, About chunk expressions

Summary

Designates a Transcript token as part of a chunk expression.

Syntax

Example Code

```
get token 1 of "someFunction(a+b)" -- yields someFunction
get token 2 of "someFunction(a+b)" -- yields (
get token 3 of "someFunction(a+b)" -- yields a
get token 3 of "a+b" -- yields b
```

Comments

Use the token keyword to parse a Transcript statement.

Comments:

A token is a string of characters delimited by certain punctuation marks. The rules for deciding what a token is are as follows:

1. Each of the following characters is a token: =, +, -, *, /, [,], (,), {, }, <, >, and comma (,).
2. Each of the following characters is a token delimiter: ; (semicolon), space, return, and tab.
3. Each string of any other characters that is either separated by one of the above characters, or delimited by double quotes, is a token. The token does not include the separating character(s).

A single token can contain multiple characters, but not multiple words, items, or lines.

Note: The token chunk is implemented for the Transcript language, and probably isn't suitable for use in a general-purpose language parser.

toLower function

Synonyms
lower

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

charToNum function, ISOToMac function, macToISO function, numToChar function, toLower function

Summary

Returns a string converted to all lowercase letters.

Syntax

the toLower of stringToConvert
toLower(stringToConvert)

Example Code

```
toLower("This is a 30• test.") -- returns "this is a 30• test."  
toLower("NO, MAÐANA!") -- returns "no, maþana!"
```

Comments

Use the toLower function to change the case of a string.

Parameters:

The stringToConvert is any string or expression that evaluates to a string.

Value:

The toLower function returns a string the same length as the stringToConvert.

Comments:

Uppercase letters, including special characters with diacritical marks, are converted to the lowercase equivalents. All other characters, including lowercase letters, numbers, punctuation, and special characters with no upper or lower case, are left unchanged by the toLower function.

Important! The toLower function supports only characters whose ASCII value is less than 128. Special characters cannot be converted by this function.

tool

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

browse keyword, brush keyword, bucket keyword, button keyword, choose command, curve keyword, dropper keyword, eraser keyword, field keyword, graphic keyword, image keyword, line keyword, newTool message, oval keyword, pencil keyword, player keyword, pointer keyword, polygon keyword, rectangle keyword, regular keyword, roundRect keyword, scrollbar keyword, select keyword, spray can keyword, tool property, Tools menu > Tools Palette

Summary

Returns the name of the currently chosen tool.

Syntax

the tool
tool()

Example Code

```
the tool  
if the tool is not "browse tool" then choose browse tool
```

Comments

Use the tool function to find out which tool is currently selected.

Value:

The tool function returns one of the following tool names:

browse, pointer, button, field, scrollbar, graphic, image, player, select, pencil, bucket, brush, eraser, spray can, rectangle, line, round rect, oval, polygon, curve, regular polygon, dropper

followed by the word `tool`.

Comments:

The tool names are the same as the names used with the choose command to select a tool for use.

tool

property

Synonyms

Objects

global, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

choose command, newTool message, tool function

Summary

Specifies which tool is currently chosen.

Syntax

set the tool [of stack] to toolName

Example Code

```
set the tool to "button tool"  
set the tool to pointer
```

Comments

Use the tool property to check which tool is in use before performing an action that depends on the tool.

Value:

The tool function returns one of the following tool names:

browse, pointer, button, field, scrollbar, graphic, image, player, select, pencil, bucket, brush, eraser, spray can, rectangle, line, round rect, oval, polygon, curve, regular polygon, dropper

followed by the word `tool`.

When setting the tool property, you can omit the word `tool`.

Comments:

The tool names are the same as the names used with the choose command to select a tool for use.

Setting the tool is equivalent to choosing the tool with the choose command.

toolTip

property

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

help message, popup command, show command, toolTipDelay property, How to get information about items on the screen

Summary

Specifies the explanatory text that pops up in a small box when the user points to an object.

Syntax

set the toolTip of object to string

Example Code

```
set the toolTip of button "OK" to "Click to accept the result"  
set the toolTip of me to field "Explanation"
```

Comments

Use the toolTip property to provide online help that explains what a control is for.

Value:

The toolTip of a control is a single line of text.

Comments:

A tool tip is a small box containing a line of text, which pops up on the screen when the mouse pointer hovers over a control. The text briefly explains the purpose and use of the control.

The tool tip appears when the mouse pointer is within the control's rectangle and when the mouse has not moved for the time specified by the toolTipDelay property. If the mouse is moving, the tool tip does not appear.

Tool tips should be short and should not attempt to explain the entire action of the control. They are best used as a short reminder or clue about what a control does.

Tool tips appear only when the Browse tool is selected.

toolTipDelay

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

toolTip property, How to disable tool tips

Summary

Specifies how long the user must point to an object with the mouse pointer before its tool tip appears.

Syntax

set the toolTipDelay to milliseconds

Example Code

```
set the toolTipDelay to 200 -- faster-than-default tooltip  
set the toolTipDelay to 0 -- no tooltip
```

Comments

Use the toolTipDelay property to control the responsiveness of tool tips.

Value:

The toolTipDelay is a non-negative integer.

By default, the toolTipDelay is 500 (half a second).

Comments:

A tool tip is a small box containing a line or two of text, which pops up on the screen when the mouse pointer hovers over a control. Use the toolTip property to set the contents of the tool tip.

The mouse must be still for the toolTipDelay time or the tool tip will not appear. If the mouse remains within a control, but is constantly moving, the toolTipDelay time does not start until the mouse stops moving.

If the toolTipDelay is zero, the tool tip is disabled and does not appear.

top
keyword

Synonyms

metacard, hypercard, supercard

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

exit to top control structure

Summary

Used in the exit to top control structure to specify the top level of execution.

Syntax

Example Code

```
exit to top
```

Comments

Use the top keyword to stop executing the current handler and all pending handlers.

Comments:

The synonyms metacard, hypercard, and supercard are included for compatibility with imported MetaCard stacks, HyperCard stacks, and SuperCard projects.

top

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

bottom property, height property, rectangle property, topLeft property, topMargin property, topRight property, Object menu > Align Selected Controls > Top, Shortcut to align control edges

Summary

Specifies how far an object's top edge is from the top of the window or screen.

Syntax

set the top of object to numberOfPixels

Example Code

```
set the top of card button 1 to 100
```

```
set the top of message window to (the bottom of this stack + 2)
```

Comments

Use the top property to change the vertical placement of a control or window.

Value:

The top of an object is an integer. A negative integer indicates that the position is above the top of the screen or card (and therefore cannot be seen).

A stack's top is the distance in pixels from the top edge of the screen to the top edge of the window's content area.

A card's or control's top is the distance in pixels from the top edge of the card to the top edge of the card or control.

Comments:

The top of a stack is in absolute (screen) coordinates. The top of a card is always zero; setting the top of a card does not cause a script error, but it has no effect. The top of a group or control is in relative (window) coordinates.

Changing the top of an object shifts it to the new position without resizing it. To change an object's height, set its height or rectangle property.

The height property of an object is equal to its bottom minus its top.

Cross-platform note: When setting the top of a window on a Mac OS or OS X system, be sure to allow 20 pixels at the top of the screen for the Mac OS menu bar.

topColor

property

Synonyms
fifthColor

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
backgroundColor property, borderColor property, bottomColor property, colorNames function, colors property, effective keyword, focusColor property, foregroundColor property, hiliteColor property, owner property, shadowColor property, threeD property, topPattern property, About colors and color references, Color Names Reference, Recipe for translating a color name to an RGB numeric triplet, Why don't buttons respect my color settings?

Summary
Specifies the color of a three-D object's raised edge.

Syntax
set the topColor of object to {empty | colorName | RGBColor}

Example Code
set the topColor of last button to 32,32,96
set the topColor of this stack to "white"
set the topColor of scrollbar ID 22 to "#339933"

Comments
Objects whose threeD property is set to true appear to stick out of or recede into the screen. Use the topColor property to specify the color of the raised edge of the object.

Value:
The topColor of an object is a color reference.

The colorName is any standard color name.

The RGBColor consists of three comma-separated integers between zero and 255, specifying the level of each of red, green, and blue; or an HTML-style color consisting of a hash mark (#) followed by three hexadecimal numbers, one for each of red, green, and blue.

By default, the topColor for all objects is empty.

Comments:

Setting the topColor of an object to empty allows the topColor of the object's owner to show through. Use the effective keyword to find out what color is used for the object, even if its own topColor is empty.

If the object's showBorder property is false, the topColor has no effect.

The setting of the topColor property has different effects, depending on the object type:

- ò The topColor of a stack determines the topColor of each object in the stack that does not have its own topColor.
- ò The topColor of a card determines the color of the border on the top and left edges of the card or group, as well as determining the topColor of each object on the card that does not have its own topColor.
- ò The topColor of a group determines the color of the border on the bottom and right edges of the group, as well as determining the topColor of each object in the group that does not have its own topColor.
- ò The topColor of a button, player, EPS object, or graphic forms a border on the top and left edges of the object. If the object's threeD property is false, the topColor has no effect.
- ò The topColor of a field forms a border on the bottom and right edges of the field and (if the field is a scrolling field) the top and left edges of the arrow boxes at the ends of the scrollbar and the bottom and right edges of the scroll area. The field's topColor also determines the color of the top and left edges of any text in the field whose textStyle is set to "threeDBox". If the field's threeD property is false, the field border is not affected.
- ò The topColor of a scrollbar forms a border on the top and left edges of the arrow boxes at the ends of the scrollbar, and the bottom and right edges of the scroll area.
- ò The topColor of an audio clip or video clip has no effect.
- ò The topColor of an image is the fifth color in the image's color palette. (To set the color of the raised edge of an image's border, set the topColor of the card, stack, or group that owns the image.)

If an object's topPattern is set, the pattern is shown instead of the color specified by the topColor.

topLeft

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomLeft property, height property, left property, location property, rectangle property, top property, topRight property, width property

Summary

Specifies the location of the specified object's upper left corner.

Syntax

set the topLeft of object to left,top

Example Code

```
set the topLeft of player 1 to 0,0
set the topLeft of last field to the bottomRight of field 1
```

Comments

Use the topLeft property to change the placement of a control or window.

Value:

The topLeft of an object is any expression that evaluates to a point—two integers separated by a comma.

The first item of the topLeft is the distance in pixels from the left edge of the screen (for stacks) or card (for all other objects) to the left edge of the object. The second item is the distance in pixels from the top edge of the screen (for stacks) or card (for all other objects) to the top edge of the object.

For cards, the topLeft property is read-only and cannot be set.

Comments:

The topLeft of a stack is in absolute (screen) coordinates. The first item (the left) of a card's topLeft property is always zero; the second item (the top) is always zero. The topLeft of a group or control is in relative (window) coordinates.

In window coordinates, the point 0,0 is at the top left of the stack window. In screen coordinates, the point 0,0 is at the top left of the screen.

Changing the topLeft of an object moves it to the new position without resizing it. To change an object's size, set its height, width, or rectangle properties.

Important! The order of the top and left parameters is reversed compared to the property name: left comes first, then top.

topLevel

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cantModify property, go command, modal command, mode property, modeless command, palette command, resizable property, sheet command, style property, About windows, palettes, and dialog boxes, Why can't I display a stack as a normal window?, Why does a stack window open in the wrong mode?, File menu > Open Stack...

Summary

Opens a stack in an editable window.

Syntax

topLevel stack

Example Code

```
topLevel stack "Internet Access"  
topLevel stack myOpenStack
```

Comments

Use the topLevel command to display a stack in an editable window.

Parameters:

The stack is any stack reference.

Comments:

An editable window can be resized (if its resizable property is true), and its controls can be selected, moved, and changed. To edit a stack of a different style (a palette, modeless, or modal dialog), use the topLevel command to display it in an editable window.

The topLevel command closes the stack and reopens it as an editable window, so closeStack and openStack, closeCard and openCard, and (if applicable) closeBackground and openBackground messages are sent to the current card as a result of executing this command. Use the lock messages command before executing topLevel if you want to prevent these messages from being sent.

If the stack is already in an editable window, the topLevel command does not close and reopen it.

topMargin

property

Synonyms

Objects

button, field, group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomMargin property, formattedHeight property, height property, leftMargin property, margins property, rightMargin property, top property

Summary

Specifies how close text within an object can come to the object's upper edge, and how close objects in a group can come to the group's upper edge.

Syntax

set the topMargin of {button | field | group} to pixels

Example Code

```
set the topMargin of button "OK" to 10
```

Comments

Use the topMargin property to change the amount of blank space between an object's top edge and its contents.

Value:

The topMargin is a non-negative integer.

By default, the topMargin of a newly created field is 8. If the field's wideMargins property is true, the field's topMargin is set to 14. The default topMargin setting for a button or group is 4.

Comments:

The topMargin property of a field specifies how many blank pixels are left between the object's top edge and its contents. The topMargin of a group specifies how far the group's top edge extends below its uppermost object.

An object's topMargin property is equal to item 2 of its margins property.

topPattern

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

backgroundPattern property, borderPattern property, bottomPattern property, brushPattern property, effective keyword, focusPattern property, foregroundPattern property, hilitePattern property, patterns property, shadowPattern property, topColor property

Summary

Specifies the pattern used to draw a three-D object's raised edge.

Syntax

set the topPattern of object to {patternNumber | imageID | empty}

Example Code

```
set the topPattern of the mouseControl to 22
set the topPattern of field "List" to the topPattern of me
```

Comments

Objects whose threeD property is set to true appear to stick out of or recede into the screen. Use the topPattern property to specify the pattern used to draw the raised edge of the object.

Value:

The topPattern of an object is a pattern specifier.

A patternNumber is a built-in pattern number between 1 and 164. (These patterns correspond to Revolution's built-in patterns 136 to 300.)

An imageID is the ID of an image to use for a pattern. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the topPattern for all objects is empty.

Comments:

Pattern images can be color or black-and-white.

Cross-platform note: To be used as a pattern on Mac OS systems, an image must be 128x128 pixels or less, and both its height and width must be a power of 2. To be used on Windows and Unix systems, height and width must be divisible by 8. To be used as a fully cross-platform pattern, both an image's dimensions should be one of 8, 16, 32, 64, or 128.

The topPattern of controls is drawn starting at the control's upper right corner: if the control is moved, the pattern does not shift.

Setting the topPattern of an object to empty allows the topPattern of the object's owner to show through. Use the effective keyword to find out what pattern is used for the object, even if its own topPattern is empty.

If the object's showBorder property is false, the topPattern has no effect.

The setting of the topPattern property has different effects, depending on the object type:

- ò The topPattern of a stack determines the topPattern of each object in the stack that does not have its own topPattern.
- ò The topPattern of a card determines the pattern of the border on the top and left edges of the card, as well as determining the topPattern of each object on the card that does not have its own topPattern.
- ò The topPattern of a group determines the pattern of the border on the bottom and right edges of the group, as well as determining the topPattern of each object in the group that does not have its own topPattern.
- ò The topPattern of a button forms a border on the top and left edges of the button. If the button's threeD property is false, the topPattern has no effect.
- ò The topPattern of a field forms a border on the bottom and right edges of the field and (if the field is a scrolling field) the top and left edges of the arrow boxes at the ends of the scrollbar and the bottom and right edges of the scroll area. The field's topPattern also determines the pattern of the top and left edges of any text in the field whose textStyle is set to "threeDBox". If the field's threeD property is false, the field border is not affected.
- ò The topPattern of a scrollbar forms a border on the top and left edges of the arrow boxes at the ends of the scrollbar, and the bottom and right edges of the scroll area.
- ò The topPattern of a graphic, image, audio clip, or video clip has no effect.
- ò The topPattern of a player or EPS object forms a border on the bottom and right edges of the object. If the object's threeD property is false, the topPattern has no effect.

If an object's topPattern is set, the pattern is shown instead of the color specified by the topColor.

topPixel

property

Synonyms
fifthPixel

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
backgroundPixel property, borderPixel property, bottomPixel property, colorMap property, focusPixel property, foregroundPixel property, hilitePixel property, screenColors function, shadowPixel property, topColor property, topPixel property, Recipe for translating a color name to an RGB numeric triplet

Summary
Specifies which entry in the color table is used for the color of a three-D object's raised edge.

Syntax
set the topPixel of object to colorNumber

Example Code
set the topPixel of player "Welcome" to 1

Comments
Use the topPixel property to change the top color of an object when the bit depth of the screen is 8 bits (256 colors) or less.

Value:
The topPixel of an object is an integer between zero and (the screenColors - 1). It designates an entry in the current color table.

By default, the topPixel for all objects is empty.

Comments:
The topPixel property specifies which entry in the color table is used for an object's top color. It is similar to the topColor property, but is specified as an entry in the current color table, rather than as a color reference.

The color table can be set by changing the colorMap property.

topRight

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomLeft property, bottomRight property, location property, rectangle property, topLeft property

Summary

Specifies the location of the specified object's upper right corner.

Syntax

set the topRight of object to right,top

Example Code

```
set the topRight of this stack to 0,20
```

Comments

Use the topRight property to change the placement of a control or window.

Value:

The topRight of an object is any expression that evaluates to a point—two integers separated by a comma.

The first item of the topRight is the distance in pixels from the left edge of the screen (for stacks) or card (for any other object) to the right edge of the object. The second item is the distance in pixels from the top edge of the screen (for stacks) or card (for any other object) to the top edge of the object.

For cards, the topRight property is read-only and cannot be set.

Comments:

The topRight of a stack is in absolute (screen) coordinates. The first item (the right) of a card's topRight property is always the width of the stack window; the second item (the top) is always zero. The topRight of a group or control is in relative (window) coordinates.

In window coordinates, the point 0,0 is at the top left of the stack window. In screen coordinates, the point 0,0 is at the top left of the screen.

Changing the topRight of an object moves it to the new position without resizing it. To change an object's size, set its height, width, or rectangle properties.

Important! The order of the top and right parameters is reversed compared to the property name: right comes first, then top.

topStack

function

Synonyms

topWindow, currentWindow

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

defaultStack property, go command, mode property, openStacks function, stacks function, topLevel command, How to bring a window to the front

Summary

Returns the long name of the topmost stack.

Syntax

the topStack

topStack()

Example Code

```
the topStack
set the defaultStack to the topStack
```

Comments

Use the topStack function to find out which stack is in the current stack.

Value:

The topStack returns the long name of a stack.

Comments:

In most applications, the active window holds the current document, and menu commands operate on the active window. In Revolution, because of the ability to open stacks in various modes, this is not necessarily the case.

Each open window has a mode associated with it. The topStack is the frontmost stack with the lowest mode.

For example, an editable window has a mode of 1, and a palette has a mode of 4. If several palettes and editable windows are open, the topStack is the frontmost editable stack, although palettes may be in front of it. If all the editable windows are then closed, the frontmost palette becomes the topStack, since there is now no window with a lower mode.

toUpper function

Synonyms
upper

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

charToNum function, ISOToMac function, macToISO function, numToChar function, toLower function

Summary

Returns a string converted to all uppercase letters.

Syntax

the toUpper of stringToConvert
toUpper(stringToConvert)

Example Code

```
toUpper("AbC; dEf") -- returns "ABC; DEF"  
toUpper("eclairs are VERY tasty!") -- returns "ECLAIRS ARE VERY TASTY!"
```

Comments

Use the toUpper function to change the case of a string.

Parameters:

The stringToConvert is any string or expression that evaluates to a string.

Value:

The toUpper function returns a string the same length as the stringToConvert.

Comments:

Lowercase letters, including special characters with diacritical marks, are converted to the uppercase equivalents. All other characters, including uppercase letters, numbers, punctuation, and special characters with no upper or lower case, are left unchanged by the toUpper function.

Important! The toUpper function supports only characters whose ASCII value is less than 128. Special characters cannot be converted by this function.

traceDelay

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

debugDo command, traceAbort property, traceReturn property, traceStack property

Summary

Used by the script debugger to specify how long to wait between lines when a handler is being traced.

Syntax

set the traceDelay to milliseconds

Example Code

```
set the traceDelay to 1000 -- 1 second between steps
```

Comments

Use the traceDelay property to make tracing faster or slower.

Value:

The traceDelay property is a positive integer.

By default, the traceDelay property is set to 500 (one-half second).

Comments:

When a handler is traced, each line is executed, after the delay specified in the traceDelay property.

Using the script debugger, you can trace a handler slowly while watching the order in which lines are executed and the value of variables as they change, in order to track down any errors in the handler.

Increasing the traceDelay causes the trace to take place more slowly, giving you more time to interpret the situation as the handler is traced. Decreasing the traceDelay speeds up tracing, which can be useful when debugging long handlers.

If the debugger is not active, this property has no effect.

Important! This property may change or be removed in future releases.

traceReturn

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

debugDo command, traceAbort property, traceDelay property, traceStack property

Summary

Used by the script debugger to specify whether a handler is being traced through, line by line.

Syntax

set the traceReturn to {true | false}

Example Code

```
set the traceReturn to true
```

Comments

Use the traceReturn property to step forward in the script being debugged.

Value:

The traceReturn property is true or false.

Comments:

When a handler is traced, each line is executed, after a specified delay. Using the script debugger, you can trace a handler slowly while watching the order in which lines are executed and the value of variables as they change, in order to track down any errors in the handler.

If the debugger is not active, setting this property has no effect.

Important! This property may change or be removed in future releases.

traceStack

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

debugDo command, traceAbort property, traceDelay property, traceReturn property

Summary

Reports the name of the stack currently being debugged.

Syntax

set the traceStack to stackName

Example Code

```
set the traceStack to "My Stack"
```

Comments

Use the traceStack property to set the stack being debugged.

Value:

The traceStack property is the name of a stack.

If the debugger is not active, the traceStack is empty.

Comments:

You can set the stack to be debugged in the Script Debug window.

Important! This property may change or be removed in future releases.

trackCount

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

enabledTracks property, mediaTypes property, tracks property

Summary

Specifies the number of separate tracks in a QuickTime movie.

Syntax

get the trackCount of player

Example Code

```
repeat with x = 1 to the trackCount of player "Arctic"  
if the trackCount of player 1 = 1 then hide player 1
```

Comments

Use the trackCount property to loop through the tracks in a QuickTime movie.

Value:

The trackCount is a positive integer.

This property is read-only and cannot be set.

Comments:

Each track of a QuickTime movie holds a different set of data, which may consist of sound, video, or other data types. The trackCount property specifies how many tracks there are.

tracks

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

enabledTracks property, mediaTypes property, trackCount property

Summary

Lists all the tracks in a QuickTime movie.

Syntax

get the tracks of player

Example Code

```
put the tracks of player myPlayerName into myNode
```

Comments

Use the tracks property to find out the contents of a QuickTime movie.

Value:

The tracks is a list of tracks, one per line. Each line consists of four items, separated by commas:

 ò the track ID (an integer)

 ò the track media type (for example, ôaudioö, ôvideoö, or ôVR Panoramaö)

This property is read-only and cannot be set.

Comments:

A movie can contain multiple tracks intended to be played at the same time (for example, an audio and a video track), or tracks that are separate (for example, an alternative audio track). You specify which tracks are active using the enabledTracks property.

transparent

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

addMax keyword, addOver keyword, addPin keyword, adMin keyword, blend keyword, blendLevel property, clear keyword, ink property, noOp keyword, notSrcAnd keyword, notSrcAndReverse keyword, notSrcBic keyword, notSrcCopy keyword, notSrcOr keyword, notSrcOrReverse keyword, notSrcXOr keyword, reverse keyword, set keyword, srcAnd keyword, srcAndReverse keyword, srcBic keyword, srcCopy keyword, srcOr keyword, srcOrReverse keyword, srcXOr keyword, subOver keyword, subPin keyword, Shortcut to make an image transparent

Summary

Specifies one of the transfer modes that can be used with the ink property. It also specifies a style of a button or field.

Syntax

Example Code

```
set the ink of graphic "Cities" to transparent
```

Comments

Use the transparent keyword to make white portions of an object disappear.

Comments:

The ink property determines how the object's color is displayed. When the transparent mode is used, white areas of the object are made transparent, allowing whatever is underneath the object to show through.

The transparent mode can be used only on Mac OS systems. On Unix and Windows systems, objects whose ink property is set to this mode appear as though their ink were set to srcCopy.

A field or button whose style property is set to `transparent` has no border or background. Text in the field or button is shown against whatever is directly underneath the field or button, instead of against the button's backgroundColor.

Setting a button's or field's style property to `transparent` sets its `opaque` to `false`.

transpose

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

keys function, matrixMultiply function, About containers, variables, and sources of value

Summary

Returns a two-dimensional array after swapping the order of keys for each element of the array.

Syntax

`transpose(array)`

Example Code

```
transpose(myArray)  
put transpose(censusByHousehold) into censusByGroup
```

Comments

Use the transpose function to swap rows and columns in a matrix.

Parameters:

The array is a two-dimensional array variable whose elements are numbers, and whose keys are sequential numbers.

Value:

The transpose function returns an array.

Comments:

A two-dimensional array is an array whose elements have a two-part key to describe them. You can visualize such an array as a set of rows and columns: the first part of each element's key is the row number, and the second part is the column number. For example, the expression `myArray[3,2]` describes the element of `myArray` which is in the third row, second column.

The transpose function simply swaps rows for columns. In other words, for each element in the array, the corresponding element in `transpose(array)` has its two parts switched one for the other. The value in the third row, second column is moved to the second row, third column.

The transpose function is its own inverse: you can transpose a transposed array again to recover the original array.

Important! If the array has missing elements, the transpose function will fail to work. For example, an array that contains elements `myArray[1,1]`, `myArray[1,2]`, and `myArray[2,2]` cannot be transposed because the element `myArray[2,1]` is missing.

traversalOn

property

Synonyms

Objects

control

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

autoHilite property, focus command, focusedObject function, focusIn message, focusOut message, showFocusBorder property, tabGroupBehavior property, How to allow copying text from a locked field, How to select text in a field

Summary

Specifies whether a control can become the active (focused) control.

Syntax

set the traversalOn of object to {true | false}

Example Code

```
set the traversalOn of group "Arrows" to true
set the traversalOn of button "Help" to false
```

Comments

Use the traversalOn property to control whether the user can tab or click into a control.

Value:

A control's traversalOn property is true or false.

Comments:

Setting a field's traversalOn to true enables the user to tab into or click in the field for editing (if the field's lockText property is false). If a field's traversalOn and lockText properties are both set to true, the user can select text, but not change it, and can scroll within the field using the keyboard. If the traversalOn is true and the lockText is false, the field can be edited. If the lockText is true and the traversalOn is false, the user can neither select nor edit the field's text.

If the object is a group, setting its traversalOn to true causes tabbing into the group to set the focus to the first control in the group.

If the object is a control other than a group or field, if its `traversalOn` is true, and if the `lookAndFeel` is set to `ôMotifö` or `ôWindows 95ö`, the user can tab to the control, then press Return or Enter to send a `mouseUp` message to the object.

If the `lookAndFeel` is set to `ôMotifö`, `ôMacintoshö`, or `ôAppearance Managerö`, the active (focused) control is outlined, and the control receives any keystrokes and the messages associated with them. (The outline can be turned off by setting the field's `showFocusBorder` property to false.) If the `lookAndFeel` is set to `ôWindows 95ö`, the appearance of a focused field does not change, but it receives keystroke messages.

Important! If an object's script uses the text selection, make sure to set the object's `traversalOn` property to false, since clicking an object whose `traversalOn` is true deselects any text selection.

true
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, false constant, not operator

Summary

Equivalent to the string `true`.

Syntax

Example Code

```
set the cantDelete of this stack to true  
if (a and b) or (c and d) is true
```

Comments

Use the true constant to indicate that a property is turned on or that a logical expression evaluates to true.

Comments:

In many cases, the true constant can be omitted without changing the meaning of the statement. For example,

```
if the lockText of field 1 is true  
is equivalent to  
if the lockText of field 1
```

trunc

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

abs function, div operator, mod operator, numberFormat property, round function, statRound function, Recipe for rounding a number up to a ceiling, Recipe for word-wrapping text to a line length

Summary

Returns the integer part of a number.

Syntax

the trunc of number
trunc(number)

Example Code

```
trunc(33.567) -- returns 33  
trunc(-6.3) -- returns -6
```

Comments

Use the trunc function to round a number down.

Parameters:

The number is any number, or expression that evaluates to a number.

Value:

The trunc function returns an integer.

Comments:

The trunc function is different from round in that truncation completely ignores the fractional part of the number. For example, round(4.9) returns 5, but trunc(4.9) returns 4. This means that the trunc function does not always return the integer closest to the number.

try
control structure

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

catch keyword, end try keyword, finally keyword, lockErrorDialogs property, throw control structure, How to find out whether a command or function is defined, Recipe for speaking an alert message

Summary

Executes a list of statements, sending any errors to the try structure to be handled.

Syntax

```
try
  statementList
catch errorVariable
  errorStatementsList
[finally
  cleanupStatementsList]
end try
```

Example Code

Comments

Use the try control structure to execute a series of statements and handle any error messages in the catch section, instead of letting Revolution display the error window.

Form:

The try control structure begins with the word try on a single line, followed by a list of Transcript statements.

The catch section begins with the catch keyword, followed by a parameter name. If any errors occur during execution of the statementList, the error message is placed in the errorVariable parameter, and the errorStatementsList is executed. Statements in the errorStatementsList can refer to the value of the errorReport.

The catch section may be followed by an optional finally section. The cleanupStatementsList is executed normally, even if Transcript encountered an exit or pass statement in the statementList.

The try structure ends with an end try statement.

Parameters:

The statementList, errorStatementsList, and cleanupStatementsList each consist of one or more valid Transcript statements.

The errorVariable is a valid variable name.

Comments:

Each of the statements in the statementList is executed normally, just as though the statementList were not in a try structure, except that any errors that would normally display an error window instead trigger the catch section. This happens even if the error is in another handler thatÆs called from within the try structure.

The statements in the catch section are executed only if an error occurs. Only errors that would normally display the error window are handled by the catch section. If the error would not normally display the error window—for example, errors when opening a file with the open file command—it doesnÆt trigger the catch section.

The statements in the finally section are executed whether or not there is an error. Since the finally section is always executed even if the statementList contains an exit or pass statement, it can be used for final cleanup actions such as deleting variables. The finally section is an optional part of the try structure.

Note: The try control structure is implemented internally as a command and appears in the commandNames.

twelveHourTime

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

convert command, time function

Summary

Specifies whether the time function uses a 12-hour clock or a 24-hour clock.

Syntax

set the twelveHourTime to {true | false}

Example Code

```
set the twelveHourTime to false
set the twelveHourTime to (the hilite of button "12 Hours")
```

Comments

Use the twelveHourTime property to operate with military-style 24-hour time.

Value:

The twelveHourTime is true or false.

By default, the twelveHourTime is true.

Comments:

The setting of the twelveHourTime property affects the format in which the time property reports a time. It also affects the format of the convert command when a time specifier is used.

If the twelveHourTime is true, the time function returns a value including AM or PM. If the twelveHourTime is false, AM or PM is not included; instead, the hour is not reset to 1 after noon. For example, the time 2:35 PM in 12-hour time is equivalent to 14:35 in 24-hour time.

two
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

constant command, second keyword

Summary

Equivalent to the number 2.

Syntax

Example Code

```
go to card two -- same as "go to card 2"
```

Comments

Use the two constant when it is easier to read than the numeral 2.

type
command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

click command, focus command, put command, rawKeyDown message, rawKeyUp message, select command, typingRate property, How to specify a character thatÆs not on the keyboard, Recipe for limiting the number of characters in a field

Summary

Simulates typing on the keyboard.

Syntax

type string [with key [,key [,key]]]

Example Code

```
type "ABC"  
type field "Example"  
type allKeyboardKeys with shiftKey,optionKey
```

Comments

Use the type command to show text appearing in a field at typing pace or to send keyboard events to a stack.

Parameters:

The string is any expression that evaluates to a string.

The key is one of commandKey, controlKey, optionKey, or shiftKey. You can specify up to three keys, separated by commas. (On Windows and Unix systems, commandKey indicates the Control key.)

Comments:

The type command sends a rawKeyDown and rawKeyUp message to the current card for each character typed.

The type command does not automatically place an insertion point in a field, so if you want to simulate typing into a field, you must first use the select command to place the insertion point:

select after field "Data"
type "Hello"

The speed at which characters are typed is controlled by the `typingRate` property. To quickly put text into a field without delay, use the `put` command instead.

Note: To type text into a field, you must first place the insertion point in the field using the `select` or `click` command.

typingRate

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

blinkRate property, idleRate property, repeatRate property, type command

Summary

Specifies a delay between characters entered by the type command.

Syntax

set the typingRate to milliseconds

Example Code

```
set the typing rate to 1000 -- 1 character per second
```

Comments

Use the typingRate property to make the type command operate faster or more slowly.

Value:

The typingRate is an integer between zero and 65535.

By default, the typingRate is 100 (one-tenth of a second).

uInt1

keyword

Synonyms

uInt1s

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

int1 keyword, read from file command, read from process command, read from socket command, uInt2 keyword, uInt4 keyword

Summary

Used with the read from file, read from process, and read from socket commands to signify a chunk of binary data the size of an unsigned 1-byte integer.

Syntax

Example Code

```
read from file myFile for 3 uInt1
```

Comments

Use the uInt1 keyword to read data from a binary file.

Comments:

If you specify uInt1 as the chunk type in a read from file, read from process, or read from socket command, the data is returned as a series of numbers separated by commas, one for each unsigned integer read.

uInt2

keyword

Synonyms

uInt2s

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

int2 keyword, read from file command, read from process command, read from socket command, uInt1 keyword, uInt4 keyword

Summary

Used with the read from file, read from process, and read from socket commands to signify a chunk of binary data the size of an unsigned 2-byte integer.

Syntax

Example Code

```
read from file "/etc/datoids" for 1 uInt2
```

Comments

Use the uInt2 keyword to read data from a binary file.

Comments:

If you specify uInt2 as the chunk type in a read from file, read from process, or read from socket command, the data is returned as a series of numbers separated by commas, one for each unsigned integer read.

uInt4

keyword

Synonyms

uInt4s

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

int4 keyword, read from file command, read from process command, read from socket command, real4 keyword, uInt1 keyword, uInt2 keyword

Summary

Used with the read from file, read from process, and read from socket commands to signify a chunk of binary data the size of an unsigned 4-byte integer.

Syntax

Example Code

```
read from file it for 17 uInt4
```

Comments

Use the uInt4 keyword to read data from a binary file.

Comments:

If you specify uInt4 as the chunk type in a read from file, read from process, or read from socket command, the data is returned as a series of numbers separated by commas, one for each unsigned integer read.

umask

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

create folder command, files function, folders function, open file command

Summary

Specifies the Unix access permissions of files and folders created by the application.

Syntax

set the umask to permissionsString

Example Code

```
set the umask to 002 -- owner read/write, anyone can read
```

Comments

Use the umask command to set the access permissions for files and folders you create.

Value:

The umask is a positive integer of three digits, or empty.

By default, the umask is set to the user's Unix `umask` setting.

Comments:

The umask blocks the specified permissions from being granted for newly created files and folders. It affects files created with the open file command, folders created with the create folder command, and files and folders created on the local system with the URL keyword.

Each digit of the umask property is an octal number that specifies a set of permissions:

- ò Read permission (4) lets a user read or copy the file or folder.
- ò Write permission (2) lets a user change the contents of the file or folder.
- ò Execute permission (1) lets a user run the file (if it is a program file), or work with files in the folder.

You add the numbers corresponding to each permission together to get the digit. For example, to specify that write permission is to be blocked, use the digit 2. To specify that read and execute permission should both be blocked, use the digit 5 (4 + 1).

The first digit of the umask specifies the permissions to be blocked for the owner of the file or folder; the second digit specifies the permissions to be blocked for members of the group that owns the file; and the third digit specifies permissions to be blocked for all other users. For example, if the umask is 022 when Revolution creates a file, the file owner has all the normal permissions, but the group and all other users do not have write permission, even if Revolution would normally create the file so as to give them write permission.

On Mac OS and Windows systems, the umask property has no effect and always reports zero.

undefine

command

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

customKeys property, define command

Summary

Has no effect and is included in Transcript for compatibility with imported Oracle Media Objects and SuperCard projects.

Syntax

undefine propertyName of object

Example Code

Comments

In SuperCard and Oracle Media Objects, the undefine command is used to release a previously-declared custom property.

In Revolution, you can use a custom property without creating it first.

underline

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bold keyword, italic keyword, plain keyword, printTextStyle property, strikeouts keyword, textStyle property, Text menu > Underline

Summary

Used with the textStyle property to indicate underlined text.

Syntax

Example Code

```
set the textStyle of field "Manhunt" to underline
```

Comments

Use the underline keyword to emphasize text by underlining it.

Comments:

You can underline an object (which underlines all the text displayed by the object) or a chunk in a field or button.

To add underlining to an object without removing other styles (such as bold), append the underline keyword to the end of the textStyle. The following example adds underlining to the styles of a button:

```
if the textStyle of button 1 is empty then
  set the textStyle of button 1 to "underline"
else
  set the textStyle of button 1 to

  (the textStyle of button 1) & comma & "underline"
end if
```

underlineLinks

property

Synonyms

showGroups

Objects

global, stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

clickText function, hide groups command, link keyword, linkColor property, show groups command, textStyle property

Summary

Specifies whether grouped text is underlined.

Syntax

set the underlineLinks to {true | false}

set the underlineLinks of stack to {true | false | empty}

Example Code

```
set the underlineLinks to (not the underlineLinks)
```

```
set the underlineLinks of this stack to false
```

```
set the underlineLinks of stack myStack to empty
```

Comments

Use the underlineLinks property to see which text is grouped.

Value:

The underlineLinks is true or false. The underlineLinks of a stack is true, false, or empty.

By default, the underlineLinks property is set to true. The underlineLinks property of a newly created stack is set to empty by default.

Comments:

When the underlineLinks property is false, grouped text looks like any other text. When the underlineLinks is true, grouped text is underlined.

If the underlineLinks of a stack is empty, grouped text in that stack is underlined or not, depending on the value of the global underlineLinks property. If the underlineLinks of a stack is true, grouped text in the stack is underlined regardless of the global setting. If the underlineLinks of a stack is false, grouped text in the stack is not underlined regardless of the global setting.

Text with its `textStyle` set to `link` is treated specially by the `clickText`, `clickChunk`, `mouseText`, and `mouseChunk` functions: a style run of grouped text is treated as a single word. This makes grouped text handy to use for hypertext or "clickable text" features.

Changes to Transcript:

The `underlineLinks` synonym and the ability to apply this property to individual stacks was introduced in version 1.1. In previous versions, the synonym `showGroups` was used, grouped text was displayed with a heavy gray underline instead of an ordinary underline when the `showGroups` was true, and the `showGroups` was set to false by default.

undo

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

clone command, copy command, create command, cut command, delete command, paste command, place command, remove command, select command, undoChanged message, Edit menu > Undo

Summary

Reverses the last user action.

Syntax

undo

Example Code

```
undo  
if it is "No! Stop! Go Back!" then undo
```

Comments

Use the undo command to give the user an opportunity to reverse a horrible mistake.

Parameters:

The undo command undoes the last action performed by the user. Undoable actions include:

- ò Using the paint tools
- ò Deleting controls by selecting them with the Pointer tool and pressing Delete
- ò Moving controls with the Pointer tool
- ò Editing actions (such as typing, cutting, and pasting) in a field

The undo command cannot be used to undo actions performed by a script. Only user actions can be undone. For example, if you use a script to change the text in a field, the undo command cannot reverse that change, but if the user edits text, you can use undo to reverse the action.

If you use the undo command twice in succession, the second undo undoes the first one.

The undo command is equivalent to choosing Edit menu Undo.

undoChanged

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

undo command

Summary

Sent to the current card when the Undo action changes.

Syntax

undoChanged

Example Code

```
on undoChanged
  global lastFieldState
  put field "Edit" into lastFieldState
end undoChanged
```

Comments

Handle the undoChanged message if you want to intercept the "Undo" menu item, providing your own Undo stack.

Comments:

Undoable actions include painting actions, deletion of objects, moving of controls, and editing actions in a field. When the user performs one of these actions, thus changing what happens as a result of choosing Edit menu Undo, Revolution sends the undoChanged message to the current card.

undoKey

message

Synonyms

Objects

control, card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

copyKey message, cutKey message, pasteKey message

Summary

Sent when the user presses the key combination equivalent to the ⌘Undo menu item.

Syntax

undoKey

Example Code

```
on undoKey
  if the optionKey is down then revert -- to last saved version
  else pass undoKey
end undoKey
```

Comments

Handle the undoKey message if you want to change the normal Undo process, or prevent use of the Undo keyboard equivalent without changing the menu.

Comments:

The Revolution development environment traps the undoKey message, unless ⌘Suspend Revolution UI is turned on in the Development menu. This means that the undoKey message is not received by a stack if itÆs running in the development environment.

The undoKey message is sent when the user presses Command-Z (on Mac OS systems), Control-Z (on Windows systems), Alt-Backspace (on Unix systems), or the keyboard Undo key.

The message is sent to the active (focused) control, or to the current card if no control is focused.

ungroup

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

delete command, group command, layer property, remove command, Object menu > Ungroup Selected

Summary

Makes a group's objects into card objects, deleting the group.

Syntax

ungroup

Example Code

```
ungroup  
if "group" is in the selectedObjects then ungroup
```

Comments

Use the ungroup command to change the constituents of a group, or to turn grouped controls into card controls.

Comments:

You can change the individual items, then use the group command or the "Group Selected" menu item in the Object menu to re-group the objects, if and only if you stay on the same card until the objects are re-grouped.

If you go to another card while the objects are ungrouped, the ungrouping becomes permanent and the objects are removed from all other cards the group was on.

unhilite

command

Synonyms
dehilite

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

autoHilite property, enable command, hilite command, hilite property, select command

Summary

Sets a button's hilite property to false.

Syntax

unhilite button

Example Code

```
unhilite button "Click Me"  
unhilite button oldSetting of group "Radio"
```

Comments

Use the unhilite command to turn off the highlighting of a button and restore it to its normal unhighlighted state, or to turn off a checkbox or radio button.

Parameters:

The button is any button reference.

Comments:

The unhilite command sets the button's hilite property to false.

unicodeText

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

charToNum function, htmlText property, numToChar function, text property, textFont property, uniDecode function, uniEncode function, URL keyword, useUnicode property, How to copy styled text between fields, How to enter or display Unicode text in a field, How to find out whether text in a field is Unicode, How to import and export styled text, How to import a Unicode text file

Summary

Specifies the text in a field, represented as Unicode (double-byte characters).

Syntax

set the unicodeText of [chunk of] field to unicodeString

Example Code

```
set the unicodeText of field 1 to URL "binfile:Chinese.txt"  
set the unicodeText of line 4 of field 1 to mySavedText
```

Comments

Use the unicodeText property to import Unicode text from a file, or copy or export the contents of a field as Unicode.

Value:

The unicodeText of a field is a string of double-byte characters.

Comments:

If the field or chunk contains single-byte characters, getting the unicodeText property converts those characters to double-byte characters automatically, so the resulting string consists entirely of double-byte characters. If any characters in the field or chunk are single-byte, the character is padded with a null byte to turn it into a double-byte character.

If you set the unicodeText of a field to a unicodeString that contains multiple languages, Revolution automatically sets the textFont of double-byte characters to the appropriate Unicode font. It also automatically converts any non-Unicode text to single-byte characters by removing the null byte used as

padding. (This saves storage space, since double-byte characters take twice as much space to store as single-byte characters.)

Note: To copy or export Unicode text from a field while preserving font, style, size, and color information, use the `htmlText` property.

unIconifyStack

message

Synonyms

Objects

card

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

closeStack message, decorations property, iconic property, iconifyStack message, resizeStack message, resumeStack message, startupIconic property

Summary

Sent when a stack is un-minimized.

Syntax

unIconifyStack

Example Code

```
on unIconifyStack -- in stack script
  put return & the short name of me after field "List"

  of stack "Open Windows List"
end unIconifyStack
```

Comments

Handle the unIconifyStack message if you want to do something special when the user expands the window to normal size.

Comments:

The terminology varies depending on platform. The unIconifyStack message is sent when the user expands (Mac OS systems), un-iconifies (Unix systems), or un-minimizes (Windows systems) the stack window.

uniDecode

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

charToNum function, fontLanguage function, ISOToMac function, macToISO function, numToChar function, toLower function, toUpper function, uniEncode function, useUnicode property, How to convert between Unicode and ASCII text

Summary

Converts a string from Unicode to single-byte text.

Syntax

the uniDecode of stringToDecode

```
uniDecode(stringToDecode[,language])
```

Example Code

```
uniDecode("A" & numToChar(zero)) -- returns "A" (on PPC)
uniDecode("ABCDE") -- returns "BD" (on Intel)
uniDecode(field "JIS Input","Japanese") -- converts to JIS
```

Comments

Use the uniDecode function to convert double-byte characters to single-byte characters.

Parameters:

The stringToDecode is any string, or expression that evaluates to a string.

The language is one of the following:

ò ANSI (synonym for ôEnglishö)

ò Arabic

ò Bulgarian

ò Chinese

ò English

ò Greek

ò Hebrew

ò Japanese (Shift-JIS)

ò Korean

- ò Polish
- ò Roman
- ò Russian (Cyrillic)
- ò Thai
- ò Turkish
- ò SimpleChinese (Simplified Chinese)
- ò Ukrainian
- ò Unicode (UTF-16)
- ò UTF8
- ò w (synonym for ôUnicodeö)

Value:

If you donÆt specify a language, the uniDecode function returns the stringToDecode, with every second byte removed.

If a language is specified, the uniDecode function encodes the stringToDecode into single-byte text, using the appropriate method for the specified language.

Comments:

The uniDecode function is the inverse of the uniEncode function and removes the null bytes inserted for Unicode compatibility. In other words, it turns double-byte characters into their closest single-byte equivalent.

If the stringToDecode contains an odd number of bytes, the last byte is ignored.

Note: You can use the UTF8 encoding only with the uniDecode and uniEncode functions. You cannot set an objectÆs textFont property to use UTF-8. To display Unicode text in an object, use either ôUnicodeö or a language name as the second item of the objectÆs textFont.

Important! The format expected by the uniDecode function is processor-dependent. On ôbig-endianö processors, where the first byte is least significant (such as Intel and Alpha processors), the uniDecode function removes the second byte of each character. On ôlittle-endianö processors, where the last byte is least significant (such as PowerPC processors), the uniDecode function removes the first byte of each character.

Changes to Transcript:

The ability to encode text in Polish was added in version 2.1.1.

The ability to handle double-byte characters on ôlittle-endianö processors was added in version 2.0. In previous versions, the uniDecode function always removed the second byte of each pair of bytes, regardless of platform.

The ability to convert Unicode text into language-specific encodings was added in version 2.0. In previous versions, the uniDecode function simply removed every other byte.

uniEncode

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

charToNum function, fontLanguage function, ISOToMac function, macToISO function, numToChar function, toLower function, toUpper function, uniDecode function, useUnicode property, How to convert between Unicode and ASCII text

Summary

Converts a string from single-byte text to Unicode.

Syntax

the uniEncode of stringToEncode

uniEncode(stringToEncode[,language])

Example Code

```
uniEncode("AB") -- returns "A",null,"B",null (on Intel)
```

```
uniEncode("AB") -- returns null,"A",null,"B" (on PPC)
```

```
uniEncode(inputText,"Japanese") -- converts Shift-JIS to Unicode
```

Comments

Use the uniEncode function to convert single-byte characters to double-byte characters.

Parameters:

The stringToEncode is any string, or expression that evaluates to a string.

The language is one of the following:

ò ANSI (synonym for ôEnglishö)

ò Arabic

ò Bulgarian

ò Chinese

ò English

ò Greek

ò Hebrew

ò Japanese (Shift-JIS)

ò Korean

ò Polish
ò Roman
ò Russian (Cyrillic)
ò Thai
ò Turkish
ò SimpleChinese (Simplified Chinese)
ò Ukrainian
ò Unicode (UTF-16)
ò UTF8
ò w (synonym for ôUnicodeö)

Value:

The uniEncode function returns a Unicode string.

If you donÆt specify a language, the string has each character of stringToEncode either followed or led (depending on platform) by a null character.

If a language is specified, the uniEncode function returns the Unicode equivalent of the stringToEncode, assuming the appropriate single-byte encoding for the specified language.

Comments:

The uniEncode function is the inverse of the uniDecode function and inserts null bytes for Unicode compatibility. In other words, it turns single-byte characters into their double-byte equivalent.

Note: You can use the UTF8 encoding only with the uniDecode and uniEncode functions. You cannot set an objectÆs textFont property to use UTF-8. To display Unicode text in an object, use either ôUnicodeö or a language name as the second item of the objectÆs textFont.

Important! The format produced by the uniEncode function is processor-dependent. On ôbig-endianö processors, where the first byte is least significant (such as Intel and Alpha processors), the uniEncode function adds the null byte after each character. On ôlittle-endianö processors, where the last byte is least significant (such as PowerPC processors), the uniEncode function adds the null byte before each character.

Changes to Transcript:

The ability to encode text in Polish was added in version 2.1.1.

The ability to handle double-byte characters on ôlittle-endianö processors was added in version 2.0. In previous versions, the uniEncode function always added the null byte after the character, regardless of platform.

The ability to convert language-specific encodings into Unicode text was added in version 2.0. In previous versions, the uniEncode function simply added a null byte.

union

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

+ operator, add command, intersect command, About containers, variables, and sources of value

Summary

Combines two arrays.

Syntax

union array with testArray

Example Code

```
union monthlyPayments with quarterlyPayments
```

Comments

Use the union command to combine two arrays, eliminating duplicate elements.

Parameters:

The array is any array variable.

The testArray is any array variable.

Comments:

The union command combines the array and testArray. Each key of the array is checked to see whether there is already an element with that key in the testArray. If there is, that element of the array is unchanged. If not, the corresponding element of the testArray is placed in the array.

After the union command is executed, the keys of the array consists of the logical union of the keys of the original array and the keys of the testArray.

The content of individual elements of the testArray does not affect the final result. Only which elements exist in the testArray, not their content, controls which elements of the testArray are placed in the array. If the array and testArray have the same set of keys but different content in each element, the union command does not change the value of the array.

unload

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

cachedURLs function, get command, httpProxy property, load command, revUnloadSpeech command, URL keyword, URLStatus function

Summary

Removes a file that was placed in the local cache by the load command, or cancels a download or upload in progress.

Syntax

unload [URL] cachedURL

Example Code

```
unload URL "http://www.example.com"  
unload URL (line 1 of the cachedURLs)  
unload URL "ftp://ftp.example.org/newfiles/latest.txt"
```

Comments

Use the unload command to release memory used by a cached URL when you no longer need it.

Parameters:

The cachedURL is any URL in the cachedURLs or any URL whose transfer was started with the libURLftpUpload, libURLftpUploadFile, or libURLDownloadToFile command.

Comments:

If the cachedURL is not in the cache and not currently being uploaded or downloaded, the result function is set to `can't find url`.

If the cachedURL is still being downloaded with the load command, the unload command cancels the download.

The unload command can be used to cancel any non-blocking FTP or HTTP file transfer in progress, but it does not cancel file transfers that were initiated by using a URL container in an expression or by putting something into a URL.

The unload command removes the cachedURL from the URLStatus function. This includes URLs that have been uploaded with the libURLftpUpload command, as well as those that have been cached with the load command.

Note: You can use a URL with the URL keyword even after it has been unloaded from the cache, but caching speeds up access to the URL.

unlock colorMap

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

lock colorMap command, lockColorMap property, screenColors function, screenDepth function, unlock cursor command, unlock screen command

Summary

Sets the lockColorMap property to false, causing the screen to be redrawn with the new color table every time the color table is changed.

Syntax

unlock colorMap

Example Code

```
unlock colorMap  
if the screenDepth <= 4 then unlock colorMap
```

Comments

Use the unlock colorMap command to allow the color table to change to accommodate newly displayed images and movies.

Comments:

The unlock colorMap command sets the lockColorMap property to false.

When all pending handlers are finished executing, the lockColorMap property is set to false automatically.

unlock cursor

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

cursor property, lock cursor command, lockCursor property, unlock moves command, unlock screen command

Summary

Sets the lockCursor property to false, allowing the cursor shape to change normally depending on what the mouse pointer is over.

Syntax

unlock cursor

Example Code

```
unlock cursor  
if the mouseControl is field "Text" then unlock cursor
```

Comments

Use the unlock cursor command to allow Revolution to change the cursor rather than retaining the cursor you set.

Comments:

The unlock cursor command sets the lockCursor property to false. When all pending handlers are finished executing, the lockCursor property is set to false automatically.

Set the cursor property to change the cursor.

unlock error dialogs

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

allowInterrupts property, errorDialog message, lock error dialogs command, lockErrorDialogs property, unlock messages command, unlock screen command, Development menu > Suppress Errors

Summary

Sets the lockErrorDialogs property to false, allowing the error window to be displayed when an execution error occurs.

Syntax

unlock error dialogs

Example Code

```
unlock error dialogs  
if the platform is not "Windows" then unlock error dialogs
```

Comments

Use the unlock error dialogs command to allow Revolution to display the standard error window when an execution error occurs.

Comments:

The unlock error dialogs command sets the lockErrorDialogs property to false. When all pending handlers are finished executing, the lockErrorDialogs property is set to false automatically.

unlock menus

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

lock menus command, lockMenus property, lockScreen property, unlock screen command

Summary

Sets the lockMenus property to false, so the menu bar is updated when the menus are changed.

Syntax

unlock menus

Example Code

```
unlock menus  
if the lockScreen is false then unlock menus
```

Comments

Use the unlock menus command to show changes in the menu bar to the user.

Comments:

The unlock menus command sets the lockMenus property to true.

When all pending handlers are finished executing, the lockMenus property is set to false, undoing the lock menus command's action. Any changes made to the menus become visible to the user.

unlock messages

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

lock messages command, lockMessages property, unlock error dialogs command, unlock recent command, unlock screen command, Development menu > Suppress Messages

Summary

Sets the lockMessages property to false, allowing messages to be sent when navigating between cards.

Syntax

unlock messages

Example Code

```
unlock messages  
if the number of this card > 10 then unlock messages
```

Comments

Use the unlock messages command to allow messages to be sent.

Comments:

The unlock messages command sets the lockMessages property to false. When all pending handlers are finished executing, the lockMessages property is automatically set to false.

unlock moves

command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

lock moves command, lockMoves property, move command, unlock messages command, unlock screen command

Summary

Sets the lockMoves property to false, allowing object motions caused with the move command to be seen.

Syntax

unlock moves

Example Code

```
unlock moves  
if the ticks > timeToGo then unlock moves
```

Comments

Use the unlock moves command to begin all pending move commands at once.

Comments:

The unlock moves command sets the lockMoves property to false. When all pending handlers are finished executing, the lockMoves property is automatically set to false.

unlock recent command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

go command, lock recent command, lockRecent property, recentCards property, recentNames property, unlock messages command, unlock screen command

Summary

Sets the lockRecent property to false, allowing cards to be added to the recent cards list.

Syntax

unlock recent

Example Code

```
unlock recent
if the short name of this card is "Overview" then unlock recent
```

Comments

Use the unlock recent command to have cards added to the recent cards list when you navigate between cards.

Comments:

When either the user or a handler navigates to a card, that card is placed on the recent cards list, and the user can return to it with the go command or by choosing View menu Go Recent.

The unlock recent command sets the lockRecent property to false. When all pending handlers are finished executing, the lockRecent property is automatically set to false.

unlock screen

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

lock screen command, lockScreen property, unlock colorMap command, unlock cursor command, unlock menus command, unlock moves command, Why don't visual effects work?

Summary

Sets the lockScreen property to false, updating the screen and displaying any changes made since the screen was locked.

Syntax

```
unlock screen [with visual [effect] effectName [speed]  
[to finalImage] [with sound audioClip]]
```

Example Code

```
unlock screen  
unlock screen with visual effect dissolve  
unlock screen with visual barn door open to white with sound "Yelp"
```

Comments

Use the unlock screen command to allow changes to the screen appearance to be seen.

Parameters:

The effectName, speed, finalImage, and audioClip are described in the visual effect command. These three parameters operate the same way as they do with the visual effect command.

Comments:

The unlock screen command sets the lockScreen property to false. When all pending handlers are finished executing, the lockScreen property is automatically set to false.

Changes to Transcript:

The ability to use QuickTime special effects was introduced in version 1.1. In previous versions, only the built-in visual effects listed under the visual effect command were available.

unmark

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

find command, mark command, mark property, print command, sort command

Summary

Sets a card's mark property to false.

Syntax

unmark {card | all cards | cards where condition}
unmark cards by finding findString [in field]

Example Code

```
unmark this card
unmark cards where the label of button "Category Popup" is "None"
unmark cards by finding it
```

Comments

Use the unmark command in conjunction with the mark command to narrow down a list of cards for printing or for special treatment by the go and sort commands.

Parameters:

The card is any card reference.

The condition is an expression that evaluates to true or false for each card in the current stack. Any object references within the condition are treated as pertaining to each card being evaluated. For example, a reference to a field is evaluated according to that field's contents on each card.

The findString is an expression that evaluates to a string.

The field is any field reference.

Comments:

You can unmark cards either one at a time, by specifying each card to be unmarked, or in batches, by using the where condition or by finding forms of the unmark command. Unmarking additional cards does not change the mark property of cards that have already been marked or unmarked.

You can use the mark and unmark commands in succession to further narrow down the set of cards. For example, this sequence of commands selects only the cards where the user has asked for help and the request has not yet been fulfilled:

```
mark cards by finding "Help" in field "History"  
unmark cards where the hilite of button "Already Helped" is true
```

When using the by finding form, the search operates the same way the normal form of the find command does. The unmark command searches for cards that contain each word in the findString. The words in the findString must be found at the beginning of a word on the card, but the words do not need to be found together.

until

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

for keyword, forever keyword, repeat control structure, while keyword

Summary

Used with the repeat control structure to specify that the loop is to repeat until the specified condition becomes true.

Syntax

Example Code

```
repeat until the seconds >= timeToStop
```

Comments

Use the until keyword to repeat a sequence of statements until a condition becomes true.

Comments:

Revolution evaluates the condition each time the loop repeats, at the top of the loop (before the statements in the repeat structure are executed). If the condition is true, the loop is skipped and execution resumes with the statement after the end repeat statement.

If the condition is already true when Revolution reaches the repeat statement, the loop is not executed at all.

up
constant

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

altKey function, commandKey function, controlKey function, down constant, left constant, mouse function, optionKey function, right constant, shiftKey function

Summary

Equivalent to the string `ôupö`.

Syntax

Example Code

```
if the shiftKey is up then beep  
wait while the mouse is up
```

Comments

Use the up constant to indicate the state of a mouse button or modifier key, or in an arrowKey handler.

URL

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

binfile keyword, file keyword, ftp keyword, htmlText property, http keyword, load command, resfile keyword, revGoURL command, About using URLs, uploading, and downloading, How to access the Internet from behind a firewall, How to display a web page in a field, Why don't URLs work in a standalone?

Summary

Designates a container consisting of an Internet resource or local file in the form of a URL.

Syntax

Example Code

```
get URL "http://www.xworlds.com/index.html"  
put URL "binfile:/Mac HD/Files/example.gif" into image "Example Logo"  
post field "Results" to URL "http://www.example.org/current.txt"  
go stack URL "http://www.example.com/stack.rev"  
put field 3 into URL "file:test.txt"
```

Comments

Use the URL keyword to access the contents of a local file or a file accessible on the Web.

Comments:

A URL is a method of designating a file or other resource. You can use a URL like any other container. You can get the contents of a URL or use its contents in any expression. Revolution supports the following URL schemes:

http: a page from a web server

ftp: a directory or file on an FTP server

file: a text file on the local disk (not on a server)

binfile: a binary file

resfile: on Mac OS and OS X systems, the resource fork of a file

All actions that refer to a URL container are blocking: that is, the handler pauses until Revolution is finished accessing the URL. Since fetching a web page may take some time due to network lag, accessing URLs may take long enough to be noticeable to the user. To avoid this delay, use the load command (which is non-blocking) to cache web pages before you need them.

For technical information about URLs and URL schemes, see RFC 1630 at [<http://www.ietf.org/rfc/rfc1630.txt>](http://www.ietf.org/rfc/rfc1630.txt).

Important! The http and ftp keywords are part of the Internet library. To ensure that these URL types work in a standalone application, when you create your standalone, make sure the "Internet libraries" option on the Resources tab of the Distribution Builder is checked.

URLDecode

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

baseConvert function, charToNum function, httpHeaders property, macToISO function, post command, URLEncode function

Summary

Returns the decoded form of a URL that was encoded for posting to an HTTP server.

Syntax

the URLDecode of formString

URLDecode(formString)

Example Code

```
URLDecode("A+B%2F1+2") -- returns "A B/1 2"  
put URLDecode(field "Test URL") after URLToCheck
```

Comments

Use the URLDecode function to decode a URL that has been sent by another system.

Parameters:

The formString is any string, or any expression that evaluates to a string.

Value:

The URLDecode function returns the formString with plus signs `+` converted to spaces and characters in the form `%NN` converted to the ASCII equivalent of the hexadecimal number NN.

Comments:

When the URLDecode function encounters a percent sign (`%`), it treats the next two characters as hexadecimal digits. (If one of the characters is not a hexadecimal digit, it's treated as a zero.) The number is converted to its character equivalent, using the character set currently in use.

URLEncode

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

baseConvert function, charToNum function, HTTPHeaders property, macToISO function, post command, URLDecode function

Summary

Returns a string that has been transformed so that it can be posted to an HTTP server as a URL.

Syntax

the URLEncode of formString

URLEncode(formString)

Example Code

```
URLEncode("ABC123") -- returns "ABC123"
```

```
URLEncode("Test string $$") -- returns "Test+string+%24%24"
```

```
put URLEncode("http://www.example.net/document.html") into newURL
```

Comments

Use the URLEncode function to encode a URL so it can be safely posted to an HTTP server.

Parameters:

The formString is any string, or any expression that evaluates to a string.

Value:

The URLEncode function returns the formString, with spaces converted to `+` and characters other than letters and numbers converted to their hexadecimal escape representation.

Comments:

Letters and numbers (alphanumeric characters) are not transformed by the URLEncode function. The representation used for non-alphanumeric characters is a percent sign followed by two hexadecimal digits. For example, the ASCII value of the character `~` is 126; the hexadecimal equivalent of 126 is 7E. So wherever the character `~` appears in the formString, it is converted to `%7E`.

URLStatus

function

Synonyms

Objects

Internet library

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cachedURLs function, ftp keyword, http keyword, libURLErrorData function, libURLSetStatusCallback command, load command, unload command, URLDecode function, URLEncode function, How to cancel a file transfer in progress

Summary

Returns the status of uploads and downloads that were started with the load, libURLDownloadToFile, libURLftpUpload, and libURLftpUploadFile commands.

Syntax

the URLStatus of url

URLStatus(url)

Example Code

```
the URLStatus of "http://www.example.com/test.html"  
wait until URLStatus(field "URL") is "cached"  
if URLStatus(myFile) is "error" then get libURLErrorData(myFile)
```

Comments

Use the URLStatus function to check the progress of an upload or download.

Parameters:

The url is a URL, or an expression that evaluates to a URL.

Value:

The URLStatus function returns one of the following values:

• `queued`: on hold until a previous request to the same site is completed

• `contacted`: the site has been contacted but no data has been sent or received yet

• `requested`: the URL has been requested

• `loading,bytesReceived,bytesTotal`: the URL data is being received

• `uploading,bytesReceived,bytesTotal`: the file is being uploaded to the URL

• `cached`: the URL is in the cache and the download is complete

öuploadedö: the application has finished uploading the file to the URL
öerrorö: an error occurred and the URL was not transferred
ötimeoutö: the application timed out when attempting to transfer the URL
empty: the URL was not loaded, or has been unloaded

Comments:

You can check the cachedURLs function to determine whether a URL has already been downloaded. The URL is not placed in the cachedURLs until the download is complete, however, so you must use the URLStatus function to check a pending download or one that has been started but not finished.

Tip: To update a progress indicator or perform other tasks during uploads and downloads, use the libURLSetStatusCallback command to automatically send a callback message every time the URLStatus function is updated. You can then write a handler for this message that performs whatever tasks are needed.

The third item (bytesTotal) in the öloadingö or öuploadingö status report is empty if it is not possible to determine the total file size. (For example, if an FTP server does not support the SIZE command, itÆs not possible to determine the file size when downloading a file from that server.)

If an error occurs during downloading, the URLStatus function returns öerrorö. You can get the error message using the libURLErrorData function.

Important! The URLStatus function is part of the Internet library. To ensure that the function works in a standalone application, you must include this custom library when you create your standalone. In Step 3 of the Distribution Builder window, make sure the öInternet Libraryö option on the Inclusions tab is checked.

Changes to Transcript:

The URLStatus function became part of the Internet library in version 1.1. In previous versions, it was not a library function.

The queued, uploading, and uploaded values were introduced in version 1.1.1. In previous versions, file transfers to the same host were not queued, and the URLStatus function could not be used to check the progress of uploads.

userLevel

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems
Not supported on Mac OS X systems
Not supported on Windows systems
Not supported on UNIX systems

See Also:

cantModify property, choose command, edit command

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the userLevel to number

Example Code

Comments

In HyperCard, the userLevel property determines what actions the user can perform. In Revolution, there are no separate user levels.

By default, the userLevel property is set to 8. (This is greater than any of the values permitted in HyperCard for the userLevel property, so imported stacks that check whether the userLevel is greater than a certain number will continue to work without modification.)

A handler can set the userLevel to any number without causing a script error, but the behavior of the application is not changed.

userModify

property

Synonyms

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Not supported on UNIX systems

See Also:

cantModify property

Summary

Has no effect and is included in Transcript for compatibility with imported HyperCard stacks.

Syntax

set the userModify to {true | false}

Example Code

Comments

In HyperCard, the userModify property determines whether the user can make temporary changes to a locked stack. In Revolution, this property has no effect.

The userModify property is always set to true. A handler can set it to any value without causing a script error, but the actual value is not changed.

useSystemDate

property

Synonyms

Objects

local

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

convert command, date function, english keyword, system keyword, time function

Summary

Specifies whether the date and time functions should be formatted in accordance with the user's system preferences, or in the standard format used in the United States.

Syntax

set the useSystemDate to {true | false}

Example Code

```
set the useSystemDate to true
```

Comments

Use the useSystemDate property to correctly format dates and times that will be viewed by the user (for example, dates and times that are displayed in a field).

Value:

The useSystemDate is true or false.

By default, the useSystemDate property is set to false.

Comments:

Setting the useSystemDate property to true does the same thing as using the system keyword with the date or time functions. If the useSystemDate is set to true, the date and time functions return the same value as the system date and the system time, respectively. If the useSystemDate is set to false, the date and time functions return the same value as the english date and the english time, respectively.

The system format is set by the Date & Time control panel (on Mac OS systems), the Date control panel (on Windows systems), or the LANG environment variable (on Unix systems).

Since the `useSystemDate` is a local property, its value is reset to false when the current handler finishes executing. It retains its value only for the current handler, and setting it in one handler does not affect its value in other handlers it calls.

useUnicode

property

Synonyms

Objects

local

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

charToNum function, numToChar function, unicodeText property, uniDecode function, uniEncode function, How to enter or display Unicode text in a field

Summary

Specifies whether the charToNum and numToChar functions assume a character is double-byte.

Syntax

set the useUnicode to {true | false}

Example Code

```
set the useUnicode to true
```

```
set the useUnicode to (the number of items of myFont is 2)
```

Comments

Use the useUnicode property to prepare to convert between Unicode characters and their numeric values.

Value:

By default, the useUnicode property is false.

Comments:

If the useUnicode property is set to true, the numToChar and charToNum functions use double-byte characters. You can pass a number greater than 255 to the numToChar function in order to generate a double-byte character, and you can pass a double-byte character to the charToNum function.

If the useUnicode is false, the numToChar and charToNum functions use single-byte characters. Passing a double-byte character to charToNum or a number larger than 255 to numToChar will produce incorrect results if the useUnicode is false.

Since the useUnicode is a local property, its value is reset to false when the current handler finishes executing. It retains its value only for the current handler, and setting it in one handler does not affect its value in other handlers it calls.

using
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

by keyword, combine command, split command, with keyword

Summary

Used with the combine and split commands to specify the delimiters for transforming a variable between an array and a string.

Syntax

Example Code

```
combine myArray using return  
combine thisArray using comma and tab
```

Comments

Use the using keyword to designate a delimiter for the string version of an array.

Comments:

The by and with keywords are synonyms for the using keyword when used with the combine or split command.

value function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

() operator, call command, do command, merge function, About messages and the message path, How to call a custom function thatÆs not in the message path

Summary

Returns the value of an expression.

Syntax

the value of expression
value(expression[,object])

Example Code

```
the value of 22 + 3 -- returns 25  
value(myVariable) -- returns value of the contents of myVariable  
the value of the selectedLine -- returns the actual text of the line  
value("the name of me", card 1) -- returns name of card 1
```

Comments

Use the value function to evaluate an expression, or to force Revolution to evaluate an expression within a statement.

Parameters:

The expression is any expression.

The object is any object reference.

Value:

The value function returns a logical, numeric, or string value, depending on the type of expression.

Comments:

The value function can be used as an instant calculator. Any arithmetic expression can be used as the expression, and the value function returns the result of the evaluation:

```
ask "What do you want to compute?"
```

answer it && "equals" && the value of it

If you specify an object, references to me in the expression are treated as references to the object. However, other object references are treated as though the handler were in the current object's script. For example, `button 3` refers to button 3 on the current card, not the card where the object is located.

If the expression is a single string, then even if it is enclosed in quotes, Revolution attempts to evaluate its contents instead of treating it as a string literal. This means that you must be careful about string literals that contain operators such as `and`.

variableNames

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

commandNames function, constant command, constantNames function, delete variable command, functionNames function, global command, globalNames function, it keyword, local command, localNames function, params function, propertyNames function

Summary

Returns a list of all parameters, local variables, and global variables.

Syntax

the variableNames

variableNames()

Example Code

```
the variableNames
```

```
repeat with x=1 to the number of items of line 4 of the variableNames
```

Comments

Use the variableNames function to find out which sources of value are available to the current handler, or to see whether a variable name is available for use.

Value:

The variableNames function returns a value consisting of four lines:

1. Parameters passed to the current handler
2. Local variables created in the current handler
3. Local variables created in the current script but outside all handlers
4. Global variables

Within each line, the variable or parameter names are separated by commas.

version function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

buildNumber function, environment function, libURLVersion function, revAppVersion function, systemVersion function

Summary

Returns the application engine's version number.

Syntax

the version
version()

Example Code

```
the version  
if the version < 2.3 then exit startup
```

Comments

Use the version function to determine whether the application supports a particular feature.

Value:

The version function returns a positive number.

Comments:

The version number belongs to either the Revolution application (if the stack is being used with Revolution) or to the running standalone application. The engine version of a standalone is the same as the engine version of Revolution that was used to create it.

vGrid

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

hGrid property, showLines property, tabStops property

Summary

Specifies whether a field's tab stops are treated as cell boundaries.

Syntax

set the vGrid of field to {true | false}

Example Code

```
set the vGrid of the fieldTemplate to true
```

Comments

Use the vGrid property to make a field behave like a spreadsheet with a grid of cells.

Value:

The vGrid of a field is true or false.

By default, the vGrid property of newly created fields is set to false.

Comments:

If the vGrid property is true, Revolution draws a vertical line at each tab stop position in the field. The lines are drawn in the field's borderColor. This property is useful for fields that are used like a spreadsheet, with each tab stop marking a column.

Any text in a column is truncated when it reaches the right edge of the column. To show the entire contents of the column, drag over the text to select it. If text at the end of a line does not have a tab following it—that is, if it's the last column in the field—it is not truncated.

If the field's tabStops property is set to empty, the vGrid lines are drawn every 32 pixels, but the text is not truncated to individual cells.

Changes to Transcript:

The use of the `borderColor` to draw grid lines was introduced in version 2.0. In previous versions, the grid lines were drawn in the `hiliteColor`.

videoClip

object

Synonyms

vc

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

templateVideoClip keyword, videoClipPlayer property, About object types and object references, How to find out whether QuickTime is available, How to list the audio clips and video clips in a stack, File menu > Import As Control > Video File...

Summary

An object type that contains movie data.

Syntax

Example Code

```
play videoClip "Spartacus"  
delete videoClip ID 2485
```

Comments

Use the videoClip object type to display a movie that is stored in the stack, rather than in another file.

Comments:

Unlike a player, a video clip contains the movie that it displays. This increases the memory required by your stack, because the movie data is loaded into memory whenever the stack file is open. However, it prevents the movie from being accidentally separated from the stack file and lost.

Video clips can be in QuickTime, AVI, or MPEG format.

A video clip is contained in a stack. Video clips cannot contain other objects. (A video clip is not a control, since it has no user interface and cannot be owned by a card.)

The video clip object has a number of properties and messages associated with it. To see a list of messages that can be sent to a video clip as a result of user actions or internal Revolution events, open the "Transcript Language Dictionary" page of the main Documentation window, and choose "Video Clip Messages" from the Show menu at the top. To see a list of all the properties a video clip can have, choose "Video Clip Properties" from the Show menu.

videoClipPlayer

property

Synonyms

vcPlayer

Objects

global

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

dontUseQT property, movie function, play command, QTVersion function, shellCommand property, sound function

Summary

Specifies the name of the program used to play movies and sounds.

Syntax

set the videoClipPlayer to playerPath

Example Code

```
set the videoClipPlayer to "/usr/bin/xanim.new"
```

Comments

Use the videoClipPlayer property to specify the video player used on Unix systems.

Value:

The videoClipPlayer is a string consisting of the file path of a program.

Comments:

Revolution uses the program specified by the videoClipPlayer when you use the play command.

By default, on Unix systems, audio clips and video clips are played with the ôxanimö player.

visible property

Synonyms
vis

Objects
any object

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:
disable command, enable command, hide command, invisible property, lockScreen property, show command, showInvisibles property, How to peek at invisible objects in a stack

Summary
Specifies whether an object can be seen or is hidden.

Syntax
set the visible of object to {true | false}

Example Code
set the visible of stack "Palette" to false
set the visible of me to (not the visible of me)

Comments
Use the visible property to determine whether an object is hidden or not, or to hide or show an object.

Value:
The visible of an object is true or false.

Comments:
A hidden object is still present and still takes up memory, and a handler can access its properties and contents, but the user cannot see or or interact with it.

An object that cannot be seen only because itÆs behind another object is still visible, in the sense that its visible property is still true.

The visible property of grouped controls is independent of the visible property of the group. Setting a groupÆs visible property to false doesnÆt change the visible property of its controls; their visible property is still true, even though the controls cannot be seen because the group is invisible.

You can set the visible property of a card, but doing so has no effect. Cards cannot be made invisible.

The visible property is the logical inverse of the invisible property. When an object's visible is true, its invisible is false, and vice versa.

visited

property

Synonyms

Objects

button, field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

linkVisitedColor property, visitedIcon property

Summary

Specifies whether a button or grouped text has been clicked during the current session.

Syntax

set the visited of button to {true | false}

set the visited of chunk of field to {true | false}

Example Code

```
set the visited of button "Start Here" to false
if the visited of the clickChunk then go back
```

Comments

Use the visited property to determine whether the user has previously used a function or traversed a link.

Value:

The visited of a button or chunk is true or false.

By default, the visited of a button or chunk is set to false when the stack opens.

Comments:

The visited property is automatically reset to false when the stack is closed (and purged from memory).

visitedIcon

property

Synonyms

Objects

button

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

icon property, showIcon property, visited property

Summary

Specifies an image to display in a button when the button's visited property is true.

Syntax

set the visitedIcon of button to {imageID | imageName}

Example Code

```
set the visitedIcon of button "To Do List" to 5499  
set the visitedIcon of me to the ID of image "Done"
```

Comments

Use the visitedIcon property to change a button's appearance when it has been clicked during the current session.

Value:

The visitedIcon property is the ID or name of an image to use for an icon. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the visitedIcon property of newly created buttons is set to zero.

Comments:

When the button has been clicked during the current session, the visitedIcon is displayed within the button's rectangle instead of the icon (if one has been specified).

visual effect command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

answer effect command, black keyword, card keyword, dontUseQT property, dontUseQTEffects property, effectRate property, find command, go command, gray keyword, hide command, inverse keyword, normal keyword, pop command, QTEffects function, show command, unlock screen command, white keyword, How to find out whether QuickTime is available, Why don't visual effects work?, Tip: Fadeout effect using blendLevel

Summary

Adds a transition effect to navigation between cards.

Syntax

visual [effect] effectName [speed] [to finalImage] [with sound audioClip]

Example Code

```
visual effect dissolve
visual effect iris close slowly
visual effect zoom close with sound "Cheers"
visual effect wipe up to black with sound "doom.wav"
```

Comments

Use the visual effect command to add a special effect the next time there is a move to another card.

Parameters:

The effectName is either an encoded QuickTime special effect description generated by the answer effect command, or one of the following:

- ò plain
- ò dissolve
- ò checkerboard
- ò venetian blinds
- ò iris close or iris open
- ò zoom close, zoom in, zoom open, or zoom out
- ò barn door close or barn door open
- ò scroll up, scroll down, scroll right, or scroll left

- ò wipe up, wipe down, wipe right, or wipe left
- ò push up, push down, push right, or push left
- ò reveal up, reveal down, reveal right, or reveal left
- ò shrink to bottom, shrink to center, or shrink to top
- ò stretch from bottom, stretch from center, or stretch from top

The speed is one of very slow, slow, normal, fast, or very fast. If no speed is specified, the visual effect takes place at normal speed.

The finalImage is one of black, white, gray, inverse, or card.

The audioClip is a reference to an audio clip in an open stack, or the file path of a sound file. The sound is played during the transition.

Comments:

When you issue a visual effect command, it is stored to be used the next time you navigate to another card in the same window with the go, find, or pop command. Usually, you place the visual effect command immediately before these commands in a handler, like this:

```
visual effect dissolve -- sets up the effect
go to card "Index" -- effect is seen during the "go"
```

However, it is not necessary to execute the navigation command immediately; the visual effect is stored and used the next time you navigate. You can even issue a visual effect command in one handler and the navigation command in another handler. All visual effects are cleared when all pending handlers exit.

The visual effect command affects only navigation within a window. If you want to create a transition effect when moving between stacks, use the go...in window form of the go command:

```
visual effect wipe down
go stack "Index" in window "Part 2" -- replaces stack on screen
```

You can issue more than one visual effect in order to stack up several effects. All the pending visual effects are executed in the order they were issued during the card transition. This example makes the card appear to shrink and then re-expand:

```
visual effect shrink to center to black
visual effect stretch from center to card
go card "Showoff"
```

The speed of visual effects is controlled by the setting of the effectRate property.

To execute a visual effect without moving to another card, lock the screen, make any changes you want to the card's appearance, and use the unlock screen with visual effect form of the lock screen command.

Visual effects do not affect the rectangle of a player whose alwaysBuffer property is set to false. For visual effects to be seen in a player's rectangle, the player's alwaysBuffer must be true.

If either the `dontUseQT` or `dontUseQTEffects` property is set to true, only the built-in visual effects can be used, and the additional effects generated by the `answer` effect command are not available.

For technical information about QuickTime special effects, see Apple Computer's QuickTime developer documentation at <http://developer.apple.com/techpubs/quicktime/qtdevdocs/REF/refEffects.3d.htm>.

Changes to Transcript:

The ability to use QuickTime special effects was introduced in version 1.1. In previous versions, only the built-in visual effects listed in the Parameters section were available.

volumes

function

Synonyms

drives

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Not supported on UNIX systems

Note: this command works differently across platforms.

See Also:

diskSpace function, files function, folders function

Summary

Returns a list of volumes.

Syntax

the volumes

volumes()

Example Code

```
the volumes
if myDisk is among the lines of the volumes then setNewDirectory
```

Comments

Use the volumes function to find out which disks are available for use.

Value:

The volumes function returns a list of volume names, one per line.

Comments:

A volume is usually a disk, but a single disk may be partitioned into multiple volumes, each of which appears as a separate icon. In this case, each partitioned volume is listed separately by the volumes function.

Disks which are physically installed or inserted into a disk drive, but are not currently mounted, do not appear in the list returned by the volumes function.

Cross-platform note: On Mac OS and OS X systems, removable drives (such as floppy drives) that are connected, but do not contain a disk, do not appear in the list returned by the volumes function. On Windows systems, empty but connected removable drives are reported by the volumes function.

This function always returns empty on Unix systems.

vScroll

property

Synonyms
scroll

Objects
field, group, stack

Platforms
Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems
Note: this command works differently across platforms.

See Also:
formattedHeight property, hScroll property, rectangle property, scrollbarDrag message, scrollbarWidth property, vScrollbar property, Recipe for a "roll credits" effect

Summary
Specifies the vertical scroll of a field, group, or stack.

Syntax
set the vScroll of {field | group} to pixels
get the scroll of stack

Example Code
set the vScroll of group "Thumbnails" to 20

Comments
Use the vScroll property to scroll a field or group, or to check how far it's been scrolled.

Value:
The vScroll of a field, group, or stack is a non-negative integer.

By default, the vScroll property of newly created objects is set to zero.

For stacks, the vScroll property is read-only and cannot be set.

Comments:
The vScroll is the amount in pixels the object has been scrolled down. If the vScroll is zero, the object has not been scrolled.

Setting the vScroll of a field or group causes a scrollbarDrag message to be sent to the field or group.

Cross-platform note: On Mac OS and OS X systems, the menu bar appears at the top of the screen, rather than inside the stack window. If a stack's editMenus property is set to false and the stack contains a menu bar, the window is scrolled down and resized so that the menu bar group is not visible

in the window. Because of this, on Mac OS and OS X systems, the vScroll of a stack reports the amount the stack has been scrolled down to hide a menu bar group.

vScrollbar

property

Synonyms

Objects

field, group

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

formattedHeight property, height property, hScrollbar property, scrollbarWidth property, vScroll property, Recipe for adding a scrollbar to a field when the contents overflow it

Summary

Specifies whether a field or group has a vertical scrollbar.

Syntax

set the vScrollbar of {field | group} to {true | false}

Example Code

```
set the vScrollbar of field "More Info" to true
```

Comments

Use the vScrollbar property to show or hide the vertical scrollbar of a field or group.

Value:

The vScrollbar of a field or group is true or false.

By default, the vScrollbar property of newly created fields or groups is set to false. The vScrollbar of newly created scrolling fields is true.

Comments:

When the vScrollbar of a group or field is set to true, the scrollbar appears along the right edge of the group or field.

Changing the vScrollbar does not change the object's rectangle property, so if there are objects near the right edge of the group, the scrollbar may cover them.

Setting the vScrollbar of a field to true is equivalent to setting the style property of the field to `scrolling`.

wait

command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

idle message, repeat control structure, seconds function, ticks function, waitDepth function, Recipe for a scrolling text banner

Summary

Pauses a handler before executing the rest of its statements.

Syntax

```
wait {until | while} condition [with messages]
wait [for] number [seconds | ticks | milliseconds] [with messages]
wait for messages
```

Example Code

```
wait for 2 seconds
wait until the mouse is up
wait while the seconds < alarmTime
```

Comments

Use the wait command to delay a certain amount of time, or until something happens, before continuing a handler.

Parameters:

The condition is an expression that evaluates to true or false. The wait command continuously evaluates this expression while waiting.

The number is an integer. If you don't specify a unit of time, the wait command waits for number ticks.

Comments:

The wait command freezes execution for the specified amount of time, or until the specified condition has been met, or until a message has been sent.

The wait for time form waits for the specified time period. (The time elapsed during a wait command may be a few milliseconds more than the specified time, depending on the speed of the system and on the system's minimum time slice.)

If the wait..with messages form is used, Revolution continues normal processing during the wait. The current handler is frozen, but the user can start other handlers and perform other actions such as switching cards.

Tip: To complete a process after a pause while allowing Revolution to run normally until that time, you can use the send in time form of the send command instead of using the wait...with messages form of the wait command.

The wait for messages form waits until any message has been sent and the message has been handled. After any message handler runs, the handler containing the wait for messages statement resumes executing.

waitDepth

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

send command, wait command

Summary

Returns the number of nested wait commands currently pending.

Syntax

the waitDepth
waitDepth()

Example Code

```
the waitDepth  
if the waitDepth > 10 then beep
```

Comments

Use the waitDepth function to determine how many wait commands are currently awaiting completion.

Value:

The waitDepth function returns a positive integer.

Comments:

The wait with messages form of the wait command , when used in a handler, pauses that handler while allowing Revolution to execute other handlers while waiting. The waitDepth function indicates how many such statements are currently executing—that is, how many handlers are currently paused by a wait with messages statement.

If there are no paused handlers, the waitDepth function returns 1.

watch
constant

Synonyms
clock

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

busy constant, constant command, cursor property

Summary

Equivalent to the number 14.

Syntax

Example Code

```
set the cursor to watch
```

Comments

Use the watch constant to set the cursor to a wristwatch shape, suitable for telling the user to wait for completion of a lengthy operation.

Comments:

The following two statements are equivalent:

```
set the cursor to watch  
set the cursor to 14
```

However, the first is easier to read and understand in Transcript code.

Important! If you use the watch cursor or other standard cursors in your application, you must include it when you create your standalone. Make sure the "Cursors" option on the Inclusions tab in Step 3 of the Distribution Builder is checked.

weekdayNames

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

convert command, date function, dateFormat function, english keyword, internet keyword, monthNames function, system keyword

Summary

Returns a list of names of the days of the week used by the date function.

Syntax

the [long|abbr[rev[iated]]|short] [english|system|internet] weekdayNames
the [internet] [english|system] weekdayNames
weekdayNames()

Example Code

```
put the weekdayNames into field "Weekdays"  
if it is not among the lines of the system weekdayNames then beep
```

Comments

Use the weekdayNames function to let the user choose from a list of weekday names, or to display a list of days in the language of the user's system preferences.

Value:

The weekdayNames function returns a set of seven day names, one per line.

The weekdayNames form returns the full name of each day, as used in the long date.

The long weekdayNames form returns the same result as the weekdayNames form.

The abbreviated weekdayNames form returns the first three letters of the name of each day, as used in the abbreviated date.

The short weekdayNames form returns the number of each day, as used in the short date.

Comments:

For each form of the weekdayNames, if the useSystemDate property is set to true, or if you specify the system weekdayNames, the list of names is provided in the language specified by the user's system preferences. If the useSystemDate is false or you specify the english weekdayNames, the list of names is provided in English.

while
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

for keyword, forever keyword, repeat control structure, until keyword

Summary

Used with the repeat control structure to specify that the loop is to repeat as long as the specified condition remains true.

Syntax

Example Code

```
repeat while the mouse is down  
repeat while myVariable is empty
```

Comments

Use the while keyword to repeat a sequence of instructions as long as a condition continues to be true.

Comments:

Revolution evaluates the condition each time the loop repeats, at the top of the loop (before the statements in the repeat structure are executed). If the condition is false, the loop is skipped and execution resumes with the statement after the end repeat statement.

If the condition is already false when Revolution reaches the repeat statement, the loop is not executed at all.

white
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

black keyword, card keyword, gray keyword, hide command, inverse keyword, show command, visual effect command

Summary

Used with the visual effect command to show a blank white screen at the end of the visual effect.

Syntax

Example Code

```
visual effect iris open to white
```

Comments

Use the white keyword to transition to a blank white background in a sequence of visual effects.

Comments:

Visual effects can be stacked in a sequence by using several visual effect commands in succession. If the last transition ends with showing the card image, and all except the last one shows an intermediate image (such as a solid white color), the effect is enhanced. You show a solid white color at the end of a transition by using the white keyword.

This example uses a camera iris effect to transition through an intermediate blank white screen to the final image:

```
visual effect iris close to white -- from card to solid white  
visual effect iris open to card -- from white to final card  
go card "Destination"
```

whole
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

characters keyword, find command, normal keyword, string keyword, wholeMatches property, words keyword

Summary

Used with the find command to search for a string of characters consisting of entire words.

Syntax

Example Code

```
find whole "this card" in field "Comments"  
find whole "run" -- finds "run", but not "grunt" or "running"
```

Comments

Use the whole keyword to find a string.

Comments:

A word is a set of characters enclosed by spaces, tabs, or returns. When using the find command, punctuation next to a word is considered part of the word.

When used with the find command, the whole keyword finds cards that contain the specified string only if the found string consists of entire words.

wholeMatches

property

Synonyms

Objects

local

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

caseSensitive property, is among operator, itemOffset function, lineOffset function, whole keyword, wordOffset function

Summary

Specifies whether the lineOffset, wordOffset, and itemOffset functions search only for entire lines, words, or items.

Syntax

set the wholeMatches to {true | false}

Example Code

```
set the wholeMatches to true
```

Comments

Use the wholeMatches property to find lines, words, or items in a container only if what you're searching for matches an entire line, word, or item.

Value:

The wholeMatches is true or false.

By default, the wholeMatches property is set to false.

Comments:

For example, suppose line 6 of a field named "Test" is "additive". The expression

`lineOffset("add",field "Test")`

evaluates to 6 if the wholeMatches is false, because "add" is part of the line "additive". If the wholeMatches is true, this function call returns empty, because "add" does not exactly match the line "additive".

If the wholeMatches property is set to true, the expression

`lineOffset(phrase, container)`

is equivalent to the expression

phrase is among the lines of container except that the first evaluates to a line number and the second evaluates to true or false. Similarly, the wordOffset and itemOffset functions operate similarly to the is among operator if the wholeMatches property is true.

Since the wholeMatches is a local property, its value is reset to false when the current handler finishes executing. It retains its value only for the current handler, and setting it in one handler does not affect its value in other handlers it calls.

wideMargins

property

Synonyms

Objects

field

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

bottomMargin property, firstIndent property, leftMargin property, margins property, rightMargin property, topMargin property

Summary

Specifies the amount of blank space at the edges of a field.

Syntax

set the wideMargins of field to {true | false}

Example Code

```
set the wideMargins of field "Info" to true
```

Comments

Use the wideMargins property to set all the margins of a field to 20 pixels.

Value:

The wideMargins of a field is true or false.

By default, the wideMargins property of newly created fields is set to false.

Comments:

The wideMargins property sets the blank space at the edges of a field to 20 pixels.

You can use the margins property instead to set all four field margins to any value. The wideMargins property is included in Transcript for compatibility with imported HyperCard stacks.

The wideMargins property interacts with the margins: setting the margins property to 20 sets the wideMargins to true, and setting the wideMargins to true sets the margins to 20.

width

property

Synonyms

Objects

any object

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

formattedWidth property, height property, left property, lockLocation property, maxWidth property, minWidth property, rectangle property, revChangeWindowSize command, right property, screenRect function, How to determine the dimensions of an image, How to prevent the user from resizing a window, How to put different-sized cards in the same stack, Object menu > Align Selected Controls > Make Widths Equal, Shortcut to equalize widths of selected controls

Summary

The width of an object is the distance from its left edge to its right edge.

Syntax

set the width of object to numberOfPixels

Example Code

```
set the width of the target to 100
set the width of button 1 to the formattedWidth of button 1
```

Comments

Use the width property to determine how much horizontal space an object needs, or to make it wider or narrower.

Value:

The width of an object is a non-negative integer.

Comments:

The width of a card is always the same as the width of its stack window. You can set the width of a card, but doing so has no effect and doesn't change the card's width property.

If you reduce the width of a stack, some objects may end up outside the stack window. These objects are not shown; however, they are still there, and will be displayed if you make the window wide enough.

If the object's lockLocation property is false, when you change its width, it shrinks or grows from the center: the object's left and right edges both shift, while the object's location property stays the same.

If the object's lockLocation property is true, it shrinks or grows from the top left corner: the object's left edge stays in the same place, and the right edge moves.

Cross-platform note: On Mac OS and OS X systems, the maximum width of an image is 16384 divided by the screen's bit depth. (For example, if the number of colors is 8Millions, the maximum image width is 4096 pixels.)

windowBoundingRect

property

Synonyms

Objects

global

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

globalLoc function, localLoc function, maxHeight property, maxWidth property, rectangle property, screenRect function, How to create a window the same size as the screen, How to take over the screen

Summary

Specifies the rectangle in which windows may open and zoom or maximize.

Syntax

set the windowBoundingRect to left,top,right,bottom

Example Code

```
set the windowBoundingRect to 100,20,800,600
```

Comments

Use the windowBoundingRect property to ensure free space on the screen for toolbars or other information on the screen.

Value:

The windowBoundingRect consists of four integers separated by commas.

By default, the windowBoundingRect is set to the screenRect. On Mac OS systems, the default windowBoundingRect is adjusted to leave room for the menu bar. On Windows systems, the default windowBoundingRect is adjusted to leave room for the task bar.

Comments:

When the user zooms (on Mac OS systems) or maximizes (on Unix and Windows systems) the window, its zoomed or maximized size is constrained by the windowBoundingRect.

Note: Users can drag or resize windows to extend outside the windowBoundingRect, and a handler can move or resize a window so that it's outside the windowBoundingRect. This property constrains the default position of windows when they open or zoom, but does not prevent them from being manually resized or repositioned.

The `windowBoundingRect` value is checked against the window's position before the `preOpenStack` message is sent. This means that if you want a stack to extend outside the `windowBoundingRect`, you should set its `rectangle` property to the desired value in a `preOpenStack` handler. This ensures that the window is enlarged to the size you specify before it appears, instead of being resized when the `windowBoundingRect` is checked.

The value of the `windowBoundingRect` does not affect palette, modal, or modeless windows. Only stacks whose `mode` property is 1 or 2 are affected.

If a stack's `formatForPrinting` property is set to `true`, the `windowBoundingRect` is ignored when the stack is opened.

The value of the `windowBoundingRect` is not updated automatically when you change the screen resolution or when you move items such as the Windows task bar. For example, if the `windowBoundingRect` is set to 0,0,640,480, it is not changed if you change the screen resolution to 1024x768. If you change the screen settings after starting up the application, make sure the value of the `windowBoundingRect` property is still appropriate.

Changing the `windowBoundingRect` does not affect the position of windows that are already open.

windowID

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

altID property, decorations property, ID property, imagePixmapID property, pixmapID property

Summary

Reports the operating system's ID for a stack window.

Syntax

get the windowID of stack

Example Code

```
put the windowID of this stack into savedID
doWindowFlash(the windowID of stack "Examples") -- an example external
```

Comments

Use the windowID property to pass to an external that needs to manipulate the window.

Value:

The windowID of a stack is an integer.

This property is read-only and cannot be set.

Comments:

The window ID is provided by the operating system, and is unique for each window.

Some externals need to manipulate the contents of a stack window directly, and do so by referencing the ID. Such externals require you to pass the ID when you use the external.

windowManagerPlace

property

Synonyms

wmPlace

Objects

stack

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

location property, rectangle property

Summary

Specifies whether a Unix window manager can automatically place a stack's window when the stack is opened.

Syntax

set the windowManagerPlace to {true | false}

Example Code

```
set the windowManagerPlace to false
```

Comments

Use the windowManagerPlace property to determine how the stack window is placed on Unix systems.

Value:

The windowManagerPlace of a stack is true or false.

Comments:

If the windowManagerPlace property is true, the window manager uses the `wmwmö` resources `clientAutoPlaceö` and `interactivePlacementö` to set the location of the window when the stack is first opened.

If the windowManagerPlace is false, the stack's rectangle property is used to set its location.

The setting of this property has no effect on Mac OS or Windows systems.

windowShape

property

Synonyms

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

decorations property, iconic property, maskData property, style property, How to remove a window's close box, How to remove a window's title bar

Summary

Specifies an image whose mask is used as the shape of the window.

Syntax

set the windowShape of stack to {imageID | 0}

Example Code

```
set the windowShape of this stack to 974
```

Comments

Use the windowShape property to display a special-purpose window shape.

Value:

The windowShape of a stack is the ID of an image whose mask will be used as the window shape. Revolution looks for the specified image first in the current stack, then in other open stacks.

By default, the windowShape property of newly created stacks is set to zero.

Comments:

The image may be in any format that supports a 1-bit transparency mask (GIF or 8-bit PNG). The shape of this mask is used to clip the window: the mask is superimposed on the window, and any pixels that fall outside the mask are not displayed. For example, if you set a stack's windowShape to an image whose mask is a rectangle with rounded corners, the window is shown with rounded corners, and the parts of the stack in the corners are hidden.

Important! The border and title bar of a stack are not shown if the stack's windowShape is set. This means you will need to provide methods of dragging and closing the window if you want the user to be able to do these tasks.

To revert to the normal window shape, set the windowShape to zero.

with
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

without keyword

Summary

Used with several commands, usually to specify a default setting or send a callback message when the operation of the command is complete.

Syntax

Example Code

```
open socket to "www.example.net" with message "mySocketOpened"  
ask file "Please name the file:" with the short name of me  
click at 100,100 with shiftKey
```

Comments

Use the with keyword to complete a command that requires it.

Comments:

The with keyword is used with the following commands:

- ò accept, load, open socket, read from socket, write to socket: specifies a callback message.
- ò answer: specifies names of buttons.
- ò answer color, answer file, answer folder, ask, ask file, ask password: specifies a default setting.
- ò click, drag, type: specifies keys to be pressed.
- ò export, import: specifies a mask file to use.
- ò hide, show, unlock screen, visual effect: specifies a visual effect or sound effect.
- ò combine, split: specifies delimiters to use for array entries.
- ò create stack: specifies a group name to include in the new stack.
- ò filter: specifies the wildcard expression.
- ò intersect, union: specifies which array to use.
- ò launch: specifies the application to use when opening a document with an application.
- ò open printing: specifies whether to show the Print dialog box.

- ò replace: specifies the replacement string.
- ò reply, request: includes an AppleEvent keyword.
- ò send to program: specifies whether to wait for a response.
- ò wait: allows messages to be sent during the wait.

When used with the combine or split command, the with keyword is a synonym for using.

within

function

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

intersect function, is within operator, mouseControl function

Summary

Returns true if a point is within an object, false if not.

Syntax

`within(object,point)`

Example Code

```
within(scrollbar ID 3,"33,40")  
within(field "News Items",the clickLoc)
```

Comments

Use the within function to determine whether a point is inside the specified object.

Parameters:

The object is any object reference.

The point is any expression that evaluates to a point—a vertical and horizontal distance from the top left of the current stack, separated by a comma. (If the object is a stack, the distance is from the top left of the screen.)

Value:

The within function returns true or false.

Comments:

If the point is within the clickable area of the object, the within function returns true, even if another object is layered on top of the object.

If the object is a graphic, its interior is considered to be within the graphic only if the graphic's filled property is true or the graphic is selected.

If the object is an image, only pixels that contain paint are considered to be within the image. Transparent parts of the image are not within the image, although they are within the image's rectangle.

If the object is a button whose menuMode is `ôtabbedö`, only the tabs are considered to be within the button. The area below the tabs is not within the button, although it is within the button's rectangle.

If the object is a stack, the within function returns true if the point is in the stack's content area, and false if the point is in the stack's title bar or borders.

The expression
 point is within the rect of object
is equivalent to
 within(object,point)
unless the object is a graphic or image, or a tabbed button.

without
keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

move command, send to program command, with keyword

Summary

Used with the move and send to program commands to allow the handler to continue executing at once. Also used with the move command to prevent messages from being sent as a result of the move.

Syntax

Example Code

```
move graphic 3 to 100,100 without waiting  
move the target to the loc of this stack without messages  
send myMessage to program "FileMaker" without reply
```

Comments

Use the without keyword to change the way the move and send to program commands operate.

Comments:

If you use the move...without waiting form of the move command or the send...to program...without reply form of the send to program command, the current handler continues immediately.

word

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

character keyword, item keyword, line keyword, token keyword, words keyword, About chunk expressions

Summary

Designates a space-delimited or quoted string as part of a chunk expression.

Syntax

Example Code

```
get last word of the long name of this stack
```

Comments

Use the word keyword to refer to one or more words in a container.

Comments:

A word is delimited by one or more spaces, tabs, or returns, or enclosed by double quotes. A single word can contain multiple characters and multiple items, but not multiple lines.

Note: Words are delimited by double quotes ("), but not by curved quotes (ôö). Revolution does not treat curved quotes as quotes.

wordOffset

function

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

caseSensitive property, is among operator, itemOffset function, lineOffset function, offset function, wholeMatches property

Summary

Returns the number of words between the beginning of a value and an occurrence of a specified string.

Syntax

`wordOffset(wordToFind,stringToSearch[,wordsToSkip])`

Example Code

```
wordOffset("hello","Well, hello there") -- returns 2  
wordOffset("ball","baseball game") -- returns 1  
wordOffset("red","A red ball on a red table",2) -- returns 4
```

Comments

Use the wordOffset function to find which word a string occurs in.

Parameters:

The wordToFind is a single word or an expression that evaluates to a word.

The stringToSearch is a string or an expression that evaluates to a string.

The wordsToSkip is a non-negative integer. If you don't specify how many wordsToSkip, the wordOffset function does not skip any words.

Value:

The wordOffset function returns a non-negative integer.

Comments:

The value returned by the wordOffset function is the number of the word where wordToFind appears in stringToSearch. If the wordToFind is not in stringToSearch, the wordOffset function returns zero. If the

wordToFind is more than one word, the wordOffset function always returns zero, even if the wordToFind appears in the stringToSearch.

If you specify how many wordsToSkip, the wordOffset function skips the specified number of words in the stringToSearch. The value returned is relative to this starting point instead of the beginning of the stringToSearch.

For example, if the stringToSearch is "This is a test" and the wordToFind is "test", wordOffset(wordToFind,stringToSearch) returns 4. However, wordOffset(wordToFind,stringToSearch,3) returns 1, even though it is finding the same occurrence, because the first three words are skipped.

words

keyword

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

characters keyword, find command, is among operator, is not among operator, items keyword, lines keyword, number function, sort command, string keyword, whole keyword, word keyword, About chunk expressions

Summary

Used with the sort command, number function, and is among and is not among properties to designate space-delimited or quoted parts of a string. Also used with the find command to search for one or more entire words.

Syntax

Example Code

```
put (field 2 is among the words of field 1) into spellcheckMe  
find words "run" -- finds "run", but not "running" or "grunt"
```

Comments

Use the words keyword to sort or select individual words.

Comments:

A word is a set of characters enclosed by spaces, tabs, or returns.

When used with the find command, the words keyword finds cards that contain each of the specified words, though not necessarily next to each other.

write to driver command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close driver command, COM1: keyword, driverNames function, int1 keyword, int2 keyword, int4 keyword, LPT1: keyword, modem: keyword, open driver command, printer: keyword, read from driver command, real4 keyword, real8 keyword, uInt1 keyword, uInt2 keyword, uInt4 keyword, write to file command, write to process command, write to socket command, Why do some characters change when transferred between systems?, Why is there a problem with line endings?

Summary

Sends data to a device that has been opened with the open driver command.

Syntax

write value to driver deviceName

Example Code

```
write field "Command" & return to driver "/dev/cu.modem"  
write nextData to driver mySerialDev
```

Comments

Use the write to driver command to control a peripheral device or send data through it.

Parameters:

The value is any expression that evaluates to a string.

The deviceName is the name of a device driver that's installed on the system and that you have previously opened with the open driver command.

Comments:

The device to write to must be opened first with the open driver command, and the mode the device was opened in must be either write or update. If the device is not open or is open read-only, the result function is set to ôFile is not open for write.ö.

Changes to Transcript:

Support for using serial drivers with OS X systems was added in version 2.0.

write to file command

Synonyms

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

Note: this command works differently across platforms.

See Also:

close file command, EOF constant, open file command, openFiles function, put command, read from file command, stdout keyword, sysError function, write to driver command, write to process command, write to socket command, How to append data to the end of a file, Why do some characters change when transferred between systems?, Why is there a problem with line endings?

Summary

Places data in a file that has been opened with the open file command.

Syntax

write value to {file filePath | stdout} [at {start | EOF | end}]

Example Code

```
write "Testing, testing" to file "testfile"
write nextData to file theFile at 350 -- starts at char 350
write linefeed to stdout
write "ATZ" to file "modem:"
```

Comments

Use the write to file command to change the contents of a file.

Parameters:

The value is any expression that evaluates to a string.

The pathName specifies the name and location of the file you want to write to. It must be the same as the path you used with the open file command. The pathName is case-sensitive, even on platforms where file names are not case-sensitive.

If you specify the name of a serial port on Mac OS or Windows systems, Revolution writes to the specified port. The names of serial ports end in a colon (:).

The start specifies the character or byte position in the file where you want to begin writing. A positive number begins start characters after the beginning of the file; a negative number begins start characters before the end of the file.

If you specify either of the synonyms EOF or end, the write begins after the last character in the file.

If you don't specify a start, the write begins:

- ò at the position determined by the seek command, or

- ò if you haven't used the seek command, wherever the last read from file or write to file command to the file left off, or

- ò if you haven't accessed the file with read from file or write to file since it was opened, after the last character (if the file was opened in append mode) or at the first character (if the file was opened in any other mode).

Comments:

The file to write to must be opened first with the open file command, and the mode the file was opened in must be write, append, or update. If the file is not open or is open read-only, the result function is set to ôFile is not open for write.ö.

If the file was opened in write mode, the write to file command completely replaces the file contents from the start. For example, if the file originally contains ôABCö, and you write ô1ö to it, after the write to file command is executed the file contains ô1ö.

If the file was opened in update mode, if you write less data to the file than it already contains, the write to file command does not remove characters from it. For example, if the file originally contains ôABCö, and you write ô1ö to it, after the write to file command is executed the file contains ô1BCö.

If the file was opened in append mode, the write begins at the end of the file.

Important! After writing, you must close the file with the close file command.

The write to stdout form writes to the standard output (on Unix systems). The standard output is always open, so you can write to it without first opening it.

Tip: As an alternative to the open file and write to file commands, you can also use the URL keyword with the put command to change the contents of a file.

write to process command

Synonyms

Objects

Platforms

Not supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

close process command, open process command, read from process command, write to driver command, write to file command, write to socket command, Why do some characters change when transferred between systems?, Why is there a problem with line endings?

Summary

Sends input to a process that was opened with the open process command.

Syntax

write value to process processName

Example Code

```
write field "Temp" to process "/bin/logall"  
write nextTick to process it
```

Comments

Use the write to process command to send input to another program.

Parameters:

The value is any expression that evaluates to a string.

The processName is case-sensitive: uppercase and lowercase characters must be exactly as they were entered in the open process command. The process must have been opened previously with the open process command; otherwise, an error message is returned in the result.

Comments:

The process to write to must be opened first with the open process command, and the mode the process was opened in must be write or update. If the process is not running or is read-only, the result function is set to ôProcess is not open for write.ö.

If there is an error writing to the process, the result function returns the error message.

Changes to Transcript:

Support for using the write to process command on OS X systems was added in version 2.0.

write to socket command

Synonyms

Objects

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

close socket command, open socket command, read from socket command, socketError message, socketTimeout message, write to driver command, write to file command, write to process command, Why do some characters change when transferred between systems?, Why is there a problem with line endings?

Summary

Sends data to a socket.

Syntax

write value to socket socket [with message callbackMessage]

Example Code

```
write field "Outputs" to socket "127.0.0.0:21"  
write "HELO" & linefeed to socket mailSocket  
write myData to socket "127.0.0.0:345" with message "doneTransact"
```

Comments

Use the write to socket command to send data from another system via a TCP socket.

Parameters:

The value is any expression that evaluates to a string.

The socketID is the identifier (set when you opened the socket) of the socket you want to send data to. The socket identifier starts with the IP address of the host the socket is connected to, and may optionally include a port number (separated from the IP address by a colon). If there is more than one socket connected to that host and port, you can specify which socket by appending the connection name or number that was assigned when the socket was opened, separated from the port number by a vertical bar (|).

The callbackMessage is the name of a message to be sent to the current object when the write is successfully completed.

Comments:

If you specify a `callbackMessage`, as soon as the write is finished, the specified message is sent to the object whose script contains the write to socket command. The `callbackMessage` is sent with one parameter, which is the `socketID`.

If you don't specify a `callbackMessage`, the handler pauses until the write has been completed, or until the time set in the `socketTimeoutInterval` property has passed.

xExtent

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems
Not supported on Mac OS X systems
Not supported on Windows systems
Supported on UNIX systems

See Also:

boundingBox property, scaleIndependently property, xOffset property, xScale property, yExtent property

Summary

Specifies the width of an EPS object's PostScript bounding box.

Syntax

set the xExtent of EPSObject to number

Example Code

```
set the xExtent of EPS "Logo" to 108 -- 1.5 inches
```

Comments

Use the xExtent property to control the appearance of an EPS object.

Value:

The xExtent of an EPS object is a positive integer.

Comments:

An EPS object's xExtent is the width in points of the PostScript. This controls the size of the PostScript when it's printed.

The xExtent of an EPS object is equal to the third item in the object's boundingBox property.

This property is supported only on Unix systems with Display PostScript installed.

xHot

property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Supported on UNIX systems

See Also:

cursor property, hotSpot property, yHot property

Summary

Specifies the horizontal position of the hot spot when an image is being used as a cursor.

Syntax

set the xHot of image to pixels

Example Code

```
set the xHot of image ID 1023 to (the height of image ID 1023)
```

Comments

Use the xHot property to specify the horizontal position of the hot spot of an image you want to use as a cursor.

Value:

The xHot of an image is an integer between 1 and the width of the image.

By default, the xHot property of newly created images is set to 1.

Comments:

A mouse cursor, regardless of its shape, always has a single active point or hot spot. For example, the arrow cursor's hot spot is at its tip, and to select an object, you must click it with the arrow tip.

The xHot property is the horizontal distance in pixels from the left edge of the image to the hot spot point.

The xHot of an image is equal to item 1 of the image's hotSpot property.

xOffset

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems
Not supported on Mac OS X systems
Not supported on Windows systems
Supported on UNIX systems

See Also:

boundingBox property, left property, xExtent property, yOffset property

Summary

Specifies the location of the left edge of an EPS object's PostScript code.

Syntax

set the xOffset of EPSObject to number

Example Code

```
set the xOffset of EPS thisObject to 144 -- 2 inches
```

Comments

Use the xOffset property to control the placement of an EPS object on the paper when it's printed.

Value:

The xOffset of an EPS object is an integer.

Comments:

The xOffset is the distance in points from the left edge of the paper to the left edge of the PostScript object. This controls the placement of the PostScript object when it's rendered.

The xOffset of an EPS object is equal to the first item in the object's boundingBox property.

This property is supported only on Unix systems with Display PostScript installed.

xScale

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

scaleIndependently property, width property, yScale property

Summary

Specifies the ratio of the width of an EPS object's PostScript bounding box to the object's screen width.

Syntax

set the xScale of EPSObject to number

Example Code

```
set the xScale of EPS 2 to 1.0 -- no scaling
```

Comments

Use the xScale property to control an EPS object's screen appearance.

Value:

The xScale of an EPS object is a positive number.

Comments:

The number 1 indicates no scaling—that is, the EPS object's width is the same as the width specified by the PostScript code it contains. Numbers greater than one indicate that the object's width is less than the PostScript width; the screen object is scaled down. Numbers less than one indicate that the object's width is greater than the PostScript width, and the screen object is scaled up from the PostScript.

In geometric terms, $xScale = xExtent / width$.

If the object's scaleIndependently property is false, the xScale property is equal to the scale of the object.

This property is supported only on Unix systems with Display PostScript installed.

yExtent

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

boundingBox property, scaleIndependently property, xExtent property, yOffset property, yScale property

Summary

Specifies the height of an EPS object's PostScript bounding box.

Syntax

set the yExtent of EPSObject to number

Example Code

```
set the yExtent of EPS myEPS to 288 -- 4 inches
```

Comments

Use the yExtent property to control the appearance of an EPS object.

Value:

The yExtent of an EPS object is a positive integer.

Comments:

An EPS object's yExtent is the height in points of the PostScript object. This controls the size of the PostScript object when it's rendered.

The yExtent of an EPS object is equal to the fourth item in the object's boundingBox property.

This property is supported only on Unix systems with Display PostScript installed.

yHot property

Synonyms

Objects

image

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

cursor property, hotspot property, xHot property

Summary

Specifies the vertical position of the hot spot when an image is being used as a cursor.

Syntax

set the yHot of image to pixels

Example Code

```
set the yHot of image ID 245 to 7
```

Comments

Use the yHot property to specify the vertical position of the hot spot of an image you want to use as a cursor.

Value:

The yHot of an image is an integer between 1 and the height of the image.

By default, the yHot property of newly created images is set to 1.

Comments:

A mouse cursor, regardless of its shape, always has a single active point or hot spot. For example, the arrow cursor's hot spot is at its tip, and to select an object, you must click it with the arrow tip.

The yHot property is the vertical distance in pixels from the top of the image to the hot spot point.

The yHot of an image is equal to item 2 of the image's hotSpot property.

yOffset

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems
Not supported on Mac OS X systems
Not supported on Windows systems
Supported on UNIX systems

See Also:

boundingBox property, top property, xOffset property, yExtent property

Summary

Specifies the location of the top edge of an EPS object's PostScript code.

Syntax

set the yOffset of EPSObject to points

Example Code

```
set the yOffset of EPS 1 to 72 -- 1 inch
```

Comments

Use the yOffset property to control the placement of an EPS object on the paper when it's printed.

Value:

The yOffset of an EPS object is an integer.

Comments:

The yOffset is the distance in points from the top edge of the paper to the top edge of the PostScript object. This controls the placement of the PostScript object when it's rendered.

The yOffset of an EPS object is equal to the second item in the object's boundingBox property.

This property is supported only on Unix systems with Display PostScript installed.

yScale

property

Synonyms

Objects

EPS

Platforms

Not supported on Mac OS systems

Not supported on Mac OS X systems

Not supported on Windows systems

Supported on UNIX systems

See Also:

height property, scaleIndependently property, xScale property

Summary

Specifies the ratio of the height of an EPS object's PostScript bounding box to the object's screen height.

Syntax

set the yScale of EPSObject to number

Example Code

```
set the yScale of last EPS to 2 -- PS image twice the height of screen
set the yScale of EPS "Guido" to 0.25 -- one-fourth the height
```

Comments

Use the yScale property to control an EPS object's screen appearance.

Value:

The yScale of an EPS object is a positive number.

Comments:

The number 1 indicates no scaling—that is, the EPS object's height is the same as the height specified by the PostScript code it contains. Numbers greater than one indicate that the object's height is less than the PostScript height; the screen object is scaled down. Numbers less than one indicate that the object's height is greater than the PostScript height, and the screen object is scaled up from the PostScript.

In geometric terms, $yScale = yExtent / height$.

If the object's scaleIndependently property is false, the yScale property is equal to the scale of the object.

This property is supported only on Unix systems with Display PostScript installed.

zero

constant

Synonyms

none

Objects

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

constant command

Summary

Equivalent to the number 0.

Syntax

Example Code

```
if the number of fields is zero then beep
```

Comments

Use the zero constant when it is easier to read than the numeral 0.

zoom

property

Synonyms

Objects

player

Platforms

Supported on Mac OS systems
Supported on Mac OS X systems
Supported on Windows systems
Not supported on UNIX systems

See Also:

constraints property, nodes property, pan property, tilt property

Summary

Specifies the field of view of a QuickTime VR movie.

Syntax

set the zoom of player to degrees

Example Code

```
set the zoom of player "Arctic" to 45 -- medium field of view  
put the zoom of player myPlayerName into currentField
```

Comments

Use the zoom property to find out how much of a QuickTime VR movie the user can see at one time.

Value:

The zoom of a player is a number between zero and 90.

Comments:

The user can change the field of view of a QuickTime VR movie using the navigational controls in the player; a handler can change the view by setting the player's zoom property.

The zoom specifies the angle of the viewer's field of view, in degrees. (Think of a camera with a zoom lens.) As the zoom increases, more of the scene can be seen. As the zoom decreases, the edges of the scene are no longer visible, but the center of the scene is shown in greater detail. A zoom of zero corresponds to the closest detailed view of a single point; a zoom of 90 corresponds to a wide-angle view of the scene.

The zoom is limited by the player's constraints property. If you specify a zoom greater than the range permitted by the constraints, the zoom is set to the highest permitted value. If you specify a zoom less than the range permitted by the constraints, the zoom is set to the lowest permitted value.

If the player does not contain a QuickTime VR movie, its zoom property is zero.

zoomBox

property

Synonyms

maximizeBox

Objects

stack

Platforms

Supported on Mac OS systems

Supported on Mac OS X systems

Supported on Windows systems

Supported on UNIX systems

See Also:

decorations property, draggable property, maximize keyword, minimize keyword

Summary

Shows a window's zoom box.

Syntax

set the zoomBox of stack to {true | false}

Example Code

```
set the zoomBox of me to true
set the zoomBox of the templateStack to false
```

Comments

Use the zoomBox property to display the zoom box or maximize box in a window's title bar.

Value:

The zoomBox property of a stack is true or false. By default, the zoomBox of a newly created stack is set to true.

Comments:

The terminology varies depending on platform. The zoomBox property determines whether the zoom box (Mac OS and OS X systems) or maximize box (Unix and Windows systems) appears in a stack window's title bar.

Note: On OS X systems, if the zoomBox property is false, the zoom box is disabled rather than hidden.

The setting of this property affects the decorations property, and vice versa. Setting a stack's zoomBox property determines whether its decorations property includes `ômaximizeö` (or is `ôdefaultö`, for window styles that normally include a zoom box). Conversely, setting a stack's decorations property sets its zoomBox to true or false depending on whether the decorations includes `ômaximizeö` (or is `ôdefaultö`).

Setting the zoomBox property causes the stack window to flash briefly.

Changes to Transcript:

The synonym `ômaximizeBoxö` was added in version 2.1.

How to access a value in an array variable

See Also:

How to change an element in an array variable, How to create an array variable, How to find out whether a container is an array, How to store an array in a custom property, [] keyword, keys function

An array variable is a variable that contains a list of values. Each element of the list has a name, called the element's key. You specify an individual elements in an array by using the array's name, along with the key for the element in square brackets []:

```
arrayName[key]
```

You can use an array reference anywhere you would use a reference to a container. For example, the following statement puts one element of an array into a variable:

```
put myArray["this"] into myVariable
```


How to access the Internet from behind a firewall

See Also:

http keyword, HTTPProxy property

To access other systems on the Internet from behind a firewall, you use a proxy server, which is a system that passes requests and data between the corporate network and the Internet. If you try to send a request directly, instead of through the proxy server, your request may be blocked by the firewall.

Set the HTTPProxy property to the proxy server's address. (If you don't know the correct address, check with your local network administrator.) Once the HTTPProxy is set, all requests for http URLs are routed through the proxy server.

How to add a pattern to the Image Chooser

See Also:

brushPattern property, penPattern property

You add a pattern to the Image Library by creating or importing the image you want to use as a pattern into your stack.

To choose your pattern in the Colors & Patterns pane of an object's property inspector, click the box for the pattern property you want to set. In the Image Chooser window, choose "This stack" from the Image Library menu at the top of the window.

All the images in the stack appear in the list of patterns, and you can click any image to create a tiling pattern for the object.

How to add an object to a group

See Also:

Object menu > Edit Group, Object menu > New Control, create command, editBackground property, start editing command

To add an object to an existing group, either use the create command or create the object when in group-editing mode.

To add an object to a group in a handler, use the create command:

```
create button "My Button" in group "My Group"
```

When an object is created in group-editing mode, it is added to the group that's being edited. You can either use the start editing command to enter group-editing mode, or select the group and choose Object menu>Edit Group. (Objects not in the group are temporarily hidden.) You can select objects in the group, move them, delete them, and add new objects. When you're finished editing the group, choose Object menu>Stop Editing Group or use the stop editing command.

How to add custom utilities to the development environment

See Also:

How to create a code library, How to install a plugin, How to open a stack automatically when starting Revolution, Development menu > Plugins, Development menu > Plugins > Plugin Settings

You add your own custom utilities to the development environment by making them into plugins. A plugin is simply a stack with whatever code and controls are needed to implement your utility. You place the stack file in the `plugins` folder inside the Revolution folder. The next time you start up Revolution, you can choose the plugin's name from Development menu>Plugins to open the stack, or choose Development menu>Plugins>Plugin Settings to open the Plugin Settings window, where you can set the plugin to be opened automatically.

You also use the Plugin Settings window to control whether your plugin receives a message when you do certain things. For example, if you want your plugin to receive a `revCloseStack` message whenever a stack is closed, mark the `revCloseStack` checkbox in the Plugin Settings window. If you include a `revCloseStack` handler in the stack script of your plugin, it will be executed whenever a stack is closed.

How to allow copying text from a locked field

See Also:

How to copy text between fields, How to select text in a field, lockText property, traversalOn property

A field's lockText property determines whether the text in the field can be changed. The traversalOn property determines whether the user can select text in the field in order to edit it. Usually, when the lockText is set to true, the traversalOn is set to false, and vice versa. However, if the lockText and traversalOn are both true, the user can drag to select text in the field and copy the text, while not being able to change the field.

You can set the lockText and traversalOn to true in the message box or a handler, or in the Basic Properties pane of the field's property inspector.

Tip: To quickly edit the text of a locked field, use the Content pane in the field's property inspector.

How to allow editing of individual cells in a table field

See Also:

How to create a spreadsheet-style table, How to format numbers in a table field

Normally, a table field is edited like any other field, by entering text directly. You can specify that a table field's cells can be edited individually. When you click a cell, what you type goes into that cell.

To allow cell editing in a table field, in the "Table" pane of the field's property inspector, check the box labeled "Cell Editing". When you click a cell in the table field, a box appears to let you edit the cell's content.

Tip: To move between cells when cell editing is enabled, use the Tab key, Return key, and arrow keys.

How to allow multiple users to access a stack on a server

See Also:

Any number of users can open and use a stack. However, if the stack is modified by more than one user, a problem arises, because when the stack file is opened, it is loaded into memory on the user's system. When the file is saved, that image in memory is saved. This means that if two users open the same stack, one user modifies and saves it, and then the other user saves the same stack, the first user's changes will be lost because they were not yet part of the stack when the second user opened it.

One method of solving this problem is to create an empty file when the stack is opened, and remove it when the stack is closed. When the first user opens the stack, then, the file is created. An openStack handler can check whether the file already exists. If it does, the application can refuse to open the stack until the file is removed, indicating that the first user has finished.

How to animate a sprite

See Also:

Animation Builder Tutorial, Why don't animations run in my standalone?, Recipe for a scrolling text banner, Tools menu > Animation Builder, move command, play command, revGoToFramePaused command, revPlayAnimation command, revStopAnimation command

A sprite is a small picture that moves along a path to create an animation. You animate a sprite by creating an object (usually an image), then using the Animation Builder to determine how it moves. Then use the revPlayAnimation command to play the animation.

To script a sprite animation without using the Animation Builder, use the move command. For example, you can draw a curve graphic, hide it, and use it to guide the move command:

```
move image "Bird" to the points of graphic "Path"
```

Tip: To make the sprite change during animation, write a message handler to make the change. Use the Keyframe Properties section of the Animation Builder window to send the message to the object during the animation.

How to append data to the end of a file

See Also:

How to create a file, How to export data to a file, write to file command

To add data to the end of a file without overwriting its contents, you use the write for append form of the write to file command.

For example, if the file's path is in a variable called `myFilePath`, the following statements append the letter `Z` to the end of the file:

```
open file myFilePath for append
write "Z" to file myFilePath
close file myFilePath
```

How to assign a custom creator signature to a standalone

See Also:

How to assign creator and type signatures to a new file, How to change the file type of an existing file, files function, fileType property, stackFileType property

On Mac OS or OS X systems, each application incorporates a creator signature. Every Mac OS application should have a unique creator signature. (Apple maintains a registry of creator signatures on its web site at <http://developer.apple.com/dev/cftype/faq.html>.)

To assign your creator signature when building an application, choose the "Mac OS Options" tab in the Distribution Builder window and enter the four-character signature in the box labeled "Creator signature". When you build the application, the signature you specified will be assigned to it.

Caution! The Distribution Builder includes a default creator signature of "RApp". Do not use this default for applications you plan to distribute: obtain a unique signature from Apple before your final build. Using multiple applications with the same signature may cause unexpected user results.

How to assign a keyboard equivalent to a button

See Also:

How to assign a keyboard equivalent to a menu item, How to make the Return or Enter key activate a button, `acceleratorKey` property, `acceleratorModifiers` property

You add a keyboard equivalent to a button by setting the button's `acceleratorKey` and `acceleratorModifiers` properties.

When the user presses the key specified by the `acceleratorKey` while holding down the modifier key or keys in the `acceleratorModifiers`, a `mouseUp` message is sent to the button, causing its `mouseUp` handler to execute.

You can either set these properties in a handler, or use the Basic Properties pane in the button's property inspector to specify a key equivalent.

How to assign a keyboard equivalent to a menu item

See Also:

How to assign a keyboard equivalent to a button, How to change a menu's contents when it is opened, How to change the menu items in a button menu, Menu Builder Tutorial, acceleratorKey property, mnemonic property

You add a keyboard equivalent to a menu item by adding a slash (/) after the menu item's name, followed by a single character. You can either modify the menu's text property in a script to add the shortcut character, or use the Basic Properties pane in the property inspector to change the menu's contents.

Tip: If the menu is part of a menu bar, you can use the Menu Builder to edit the menu bar and add keyboard equivalents.

When the user presses the Command key (on Mac OS or OS X systems) or the Control key (on Unix or Windows systems) with the shortcut character, the menu item is chosen.

How to assign an icon to a Mac OS standalone application

See Also:

How to assign an icon to an OS X standalone application, How to assign an icon to a Windows standalone application, How to associate files with a Mac OS standalone application, How to copy a resource fork, How to respond to double-clicking a file for Mac OS or OS X, About file types, application signatures, and file ownership, File menu > Distribution Builder

To assign an icon to a standalone application, you first create the icon, then use the Distribution Builder to place the icon in the application.

For Mac OS applications, you create a set of icons with ResEdit or a similar utility, and save them in a resource file. There are six standard icon resource types: ICN# (black-and-white), icl4 (four-bit color), icl8 (8-bit color), ics# (black-and-white small), ics4 (4-bit small), and ics8 (8-bit small). Each of these application icons should have the resource ID 128. You can also include other resources you want to add to your application in this file.

In the Distribution Builder, when you reach the “Optional Settings” step, choose the Mac OS tab, click Choose, and select the file that contains the icons. The icon set is automatically added to your application and appears when your application’s file is displayed.

How to assign an icon to a Windows standalone application

See Also:

How to assign an icon to a Mac OS standalone application, How to assign an icon to an OS X standalone application, How to associate files with a Windows standalone application, How to respond to double-clicking a file for Windows, About file types, application signatures, and file ownership, File menu > Distribution Builder

To assign an icon to a standalone application, you first create the icon, then use the Distribution Builder to place the icon in the application.

For Windows applications, you create a 16-color (4-bit) .ico file. It is important that the file be saved with 16 colors.

Important! The icon file's size must be 744 bytes. If the file is not this size, it's not in the correct format, and trying to use it as an application icon won't work.

In the Distribution Builder, when you reach the "Optional Settings" step, choose the Windows tab, click Choose Application Icon, and select the icon file. The icon is automatically added to your application and appears when your application's file is displayed.

How to assign an icon to an OS X standalone application

See Also:

How to assign an icon to a Mac OS standalone application, How to assign an icon to a Windows standalone application, How to associate files with an OS X standalone application, How to respond to double-clicking a file for Mac OS or OS X, About file types, application signatures, and file ownership, File menu > Distribution Builder

To assign an icon to a standalone application, you first create the icon file, then use the Distribution Builder to place the icon in the application.

For OS X applications, you create an icon file using IconComposer or a similar utility.

In the Distribution Builder, when you reach the “Optional Settings” step, choose the OS X tab, click Choose, and select the icon file. The icon is automatically added to your application and appears when your application’s file is displayed and in the Dock when your application is running.

How to assign creator and type signatures to a new file

See Also:

How to change the file type of an existing file, How to create a file, files function, fileType property, stackFileType property

On Mac OS or OS X systems, each file incorporates a type signature and creator signature, which tell the operating system what kind of file it is and what application it belongs to. You assign a file type and creator signature to a file by setting the fileType property before saving the file. The following set of statements create a read-only text file (type: `txttro`) belonging to SimpleText (whose creator signature is `ttxttro`):

```
set the fileType to "txttro"  
open file "Release Notes"
```

Tip: To change the type signature or creator of an existing file without saving it, use AppleScript with the do command.

How to assign help text to an object

See Also:

Object menu > Object Inspector, toolTip property

You add simple help text to an object by setting the object's toolTip property.

When the user hovers over the object, its tool tip appears in a small box. The tool tip can be used to provide a quick description of the object's function.

You can either set the toolTip property in a handler, or use the Basic Properties pane in the property inspector to specify a tool tip. The following statement sets a button's toolTip:

```
set the toolTip of button "Find" to "Search for a title"
```

How to associate files with a Mac OS standalone application

See Also:

How to assign an icon to a Mac OS standalone application, How to assign creator and type signatures to a new file, How to associate files with an OS X standalone application, How to associate files with a Windows standalone application, How to change the file type of an existing file, How to create a file, How to respond to double-clicking a file for Mac OS or OS X, How to store preferences or data for a standalone application, About file types, application signatures, and file ownership, File menu > Distribution Builder, fileType property, save command, stackFileType property

When you double-click a document file, it automatically opens in the application itÆs associated with. For Mac OS systems, you associate files with an application by assigning the applicationÆs four-character creator signature to the files.

Every Mac OS and OS X application should have a unique creator signature. (Apple maintains a registry of creator signatures on its web site at <<http://developer.apple.com/dev/cftype/>>.) To assign your creator signature when building an application, enter the signature on the Mac OS tab in Step 3 of the Distribution Builder.

To associate a file with your application, before you create the file, use the stackFileType property (for stack files created with the save command) or the fileType property (for all other files) to set the fileÆs creator to the same four-character string you used for the application.

How to associate files with a Windows standalone application

See Also:

How to assign an icon to a Windows standalone application, How to associate files with a Mac OS standalone application, How to associate files with an OS X standalone application, How to create a file, How to respond to double-clicking a file for Windows, How to store preferences or data for a standalone application, About file types, application signatures, and file ownership, Recipe for registering a Windows extension, File menu > Distribution Builder, fileType property, save command, stackFileType property

When you double-click a document file, it automatically opens in the application it's associated with. On Windows systems, a filename's extension determines which application owns the file. The Windows registry holds the association between the application and the file extension.

To create the necessary registry entries for your application, you use the setRegistry function when the application is installed. (For example code, see the topic "Recipe for registering a Windows extension" in the Transcript Cookbook section of the documentation.)

Once the registry is set up to recognize a particular extension as belonging to your application, you can associate a file with the application simply by using the extension in its name:

```
save stack "Doc" as "Doc.xyz" -- your app's extension
```

How to associate files with an OS X standalone application

See Also:

How to assign an icon to an OS X standalone application, How to assign creator and type signatures to a new file, How to associate files with a Mac OS standalone application, How to associate files with a Windows standalone application, How to change the file type of an existing file, How to create a file, How to respond to double-clicking a file for Mac OS or OS X, How to store preferences or data for a standalone application, About file types, application signatures, and file ownership, File menu > Distribution Builder, fileType property, save command, stackFileType property

When you double-click a document file, it automatically opens in the application itÆs associated with. For OS X systems, you associate files with an application by assigning the applicationÆs four-character creator signature to the files.

Every Mac OS and OS X application should have a unique creator signature. (Apple maintains a registry of creator signatures on its web site at <<http://developer.apple.com/dev/cftype/>>.) To assign your creator signature to your application, enter the signature on the OS X tab in Step 3 of the Distribution Builder.

To associate a file with your application, before you create the file, use the stackFileType property (for stack files created with the save command) or the fileType property (for all other files) to set the fileÆs creator to the same four-character string you used for the application.

How to attach data to an object

See Also:

How to put text into a field, Object menu > Object Inspector, Object menu > Card Inspector, Object menu > Stack Inspector, customKeys property, customProperties property, text property

You put text into buttons and fields by setting their text property:

```
set the text of field 6 to myInfo  
put "Help" into button "Data"
```

Custom properties are a more general solution. You can create a custom property for an object of any type. Custom properties can contain any kind of data—text, numbers, binary data—and are stored with the object, just like built-in properties.

You use the Custom Properties pane in the object's property inspector to manage an object's custom properties.

How to automatically adjust objects when a window is resized

See Also:

How to prevent the user from resizing a window, Geometry Management Tutorial, Recipe for truncating text to a pixel width, location property, rectangle property, `resizeStack` message

You use the Geometry pane of an object's property inspector to automatically adjust the size and position of the object when the user resizes a window.

To automatically adjust object layout, open the object's property inspector and choose "Geometry" from the menu at the top. You can either move or reposition the object automatically.

Tip: If you want to perform custom adjustments that are too complex for the Geometry pane, create a `resizeStack` handler in the card or stack script, and write statements to properly move the objects. The `resizeStack` message is sent to the current card whenever a stack window is resized.

How to automatically color-code a script

See Also:

How to pretty-print a script, View menu > Colorization Settings... (Script Editor), View menu > Colorize Live (Script Editor)

Many programmers find it helpful to format their code so that different kinds of terms (commands, functions, and so on) are shown in different colors. This helps you quickly scan your scripts, and may also help you find problems such as misspelled Transcript terms.

You set Revolution to automatically color scripts as you type by choosing Edit menu>Preferences. In the `Script Editor` pane, check the box labeled `Colorize while typing`. You can also turn this setting on or off in the script editor by choosing View menu>Live Colorization.

Tip: Turning automatic colorizing on or off does not affect existing code, only new lines you type. To color-code a script, choose Script menu>Colorize in the script editor. To remove the colors, select the entire script, then choose Text menu>Colors>Use Default Color.

How to automatically include groups on a new card

See Also:

How to create a card template for a stack, `backgroundBehavior` property, `place` command

You arrange to have a group automatically included on all new cards by setting the group's `backgroundBehavior` property to `true`.

When you create a new card, any groups on the previous card whose `backgroundBehavior` is `true` are automatically placed on the newly created card. If you create a group of objects that serves as a template for all cards, you should set the group's `backgroundBehavior` to `true` so it will be automatically placed on new cards you create.

You can either set this property in a handler or the message box, or use the Basic Properties pane in the group's property inspector.

Tip: To add a group to a card after it's been created, use the `place` command.

How to beep

See Also:

How to make the computer speak out loud, AudioClip object, beep command, play command

You make the system beep by using the beep command.

If you want more than one beep, specify the number of beeps along with the command:

```
beep 3 -- beeps three times
```

How to block or change the action of arrow keys

See Also:

Why can't I use the arrow keys when editing a field?, Shortcut to go to the next card, Shortcut to go to the previous card, Shortcut to navigate in a field, arrowKey message, navigationArrows property, textArrows property

You use an arrowKey handler to change the way arrow keys behave. The arrowKey handler intercepts arrow key presses, and the keypress is processed only if you pass the arrowKey message.

The statements in the arrowKey handler are executed when an arrow key is pressed. If the arrowKey handler does not contain any statements, so the arrowKey message is not passed, the arrow keys do nothing:

```
on arrowKey
  -- nothing here - the message is blocked
end arrowKey
```

Tip: By default, the arrow keys move the insertion point when there is a text selection, or move from card to card otherwise. To change this behavior, set the navigationArrows and textArrows properties to false.

How to break a line in a script

See Also:

View menu > Wrap Long Script Lines (Script Editor), keyword

If a line of code is too long to be easily displayed in the script editor, it is convenient to break it into more than one line for display, while still having Transcript treat it as a single line.

You use the character to break a script line for display, as in the following example:

```
set the thumbSize of scrollbar 1 to
```

```
    (the height of group 1/the formattedHeight of group 1)
```

When the above split line is executed, it's treated as a single line of code.

How to bring a control to the front

See Also:

How to move one control in front of another, How to change the tab order of controls on a card, Object menu > Bring to Front, Object menu > Send to Back, layer property, number property

You move a control to be in front of all other controls by setting the control's layer property to `top`:

set the layer of field "Front Field" to top

The layer of a control determines its back-to-front order on the card. For example, the bottommost control's layer is 1. If two controls overlap, the one with the higher layer is in front.

You can either set this property in a handler or the message box, or use the Size & Position pane in the card's property inspector to change the control's layer. You can also select the control and choose Object menu>Bring to Front to move the control to the top.

How to bring a window to the front

See Also:

How to change the order of cards in a stack, How to display another stack in the current window, How to find out whether a window is open, How to keep a window on top of other windows, How to list all open windows, How to open a stack, Recipe for a Window menu, Window menu > Send Window to Back, defaultStack property, go command, modal command, mode property, modeless command, palette command, topLevel command, topStack function

You use the go command to bring a stack window to the front, as in the following statement:

```
go stack "My Stack"
```

If the stack is not already open, the go command opens it. If it is already open, the go command brings it to the front. (The topLevel, modal, modeless, and palette commands also bring the specified window to the front, but may also change its mode.)

Note: If the raisePalettes property is true, all palettes are always in front of editable windows and modeless dialog boxes. Using the go command to bring an editable window or modeless dialog to the front does not move it in front of palettes.

How to call a custom function that's not in the message path

See Also:

How to use a handler outside the message path, About messages and the message path, call command, insert script command, start using command, value function

You use the value function to call a function that's in the script of an object that isn't in the message path. Usually, you can only call custom functions that are somewhere in the message path, but you can use the value function to call any function in any object in an open stack.

For example, suppose you want to use a function named `myFunction` which is defined in the script of card 1 of a stack named `My Stack`. The following statement can be used to call the function from any script or from the message box:

```
get value("myFunction()",card 1 of stack "My Stack")
```

Tip: You can use the insert script command to place the object in the message path. In this case, you don't need to use the value function.

How to cancel a file transfer in progress

See Also:

How to cancel a pending event, How to download a file from an FTP server, How to stop a running handler, How to unblock upload and download operations, About using URLs, uploading, and downloading, cachedURLs function, delete URL command, load command, libURLDownloadToFile command, libURLftpUpload command, libURLftpUploadFile command, unload command, URLStatus function

You use the unload command to cancel a download or upload.

```
-- start file transfer:  
libURLDownloadToFile "ftp://example.org/new_beta",it  
-- cancel the file transfer:  
unload URL "ftp://example.org/new_beta"
```

The unload command can be used to cancel any non-blocking FTP or HTTP file transfer in progress, but it does not cancel file transfers that were initiated by using a URL container in an expression or by putting something into a URL.

Note: When using the unload command, you must specify the URL just as it was specified in the command that started the file transfer.

How to cancel a pending message

See Also:

How to cancel a file transfer in progress, How to delay an action, How to schedule a future message, How to stop a running handler, Recipe for a card slideshow, cancel command, flushEvents function, pendingMessages function, send command

If an event is scheduled with the send command but has not yet been executed, you can prevent it with the cancel command:

```
cancel savedMessageID
```

The message ID is returned in the result function when you execute the send command, so you can save it in a variable or custom property in case you want to cancel the pending message.

Tip: To cancel user actions (such as clicks) and prevent them from triggering unwanted handlers, you use the flushEvents function.

How to capture video from a video camera

See Also:

How to change the size and position of the video grabber window, How to change video grabber settings, How to take a picture with a video camera, `revInitializeVideoGrabber` command, `revRecordVideo` command, `revStopRecordingVideo` command

You record video from a video camera (or other video input) to a file using the `revRecordVideo` command.

Before recording video, you must use the `revInitializeVideoGrabber` command to start up video capture. Then use the `revRecordVideo` command to start capturing data to a file:

```
revRecordVideo "/Disk/Folder/file.mov"
```

When you're finished, use the `revStopRecordingVideo` command.

Tip: To see the input from a video camera in the video grabber without recording it to a file, use the `revPreviewVideo` command instead of `revRecordVideo`.

How to center a window on the screen

See Also:

How to close a window, How to change the size and position of a window, How to respond to moving a window, location property, rectangle property, screenLoc function, windowBoundingRect property

You center a stack window on the screen by setting its location property to the center of the screen. The center of the main screen is returned by the screenLoc function, so you can center a stack window using the following statement, in a handler or in the message box:

```
set the loc of stack "My Window" to the screenLoc
```

Tip: All windows in a Revolution application—editable window, modal dialog box, modeless dialog box, or palette—are stacks, and any of them can be moved to any place on the screen by changing their location property.

How to change a menu's contents when it is opened

See Also:

About menus and the menu bar, How to change the menu items in a button menu, How to change the selected item in an option menu or combo box, Recipe for automatically updating a Text menu, Recipe for automatically updating a Window menu, Recipe for checkmarking a menu item, mouseDown message, put command, text property

You change a menu's contents by changing the contents of the button associated with the menu. To change a menu when the user opens it—for example, to change the wording of a menu item depending on whether the Shift key is held down—use a mouseDown handler in the button's script:

```
on mouseDown
  if the shiftKey is down then
    put "Paste Special" into line 5 of menu "Edit"
  else put "Paste" into line 5 of menu "Edit"
end mouseDown
```

Cross-platform note: On Mac OS and OS X systems, when opening a menu-bar menu, the mouseDown message is sent to the button's group, not to the button itself. Place the handler in the menubar group's script.

How to change a script at run time

See Also:

How to edit an object's script, How to execute a script statement dynamically, How to execute Transcript code in a file, Recipe for setting a script to the contents of a field, script property

You change an object's script by setting its script property.

Note: Because the script of an object is a property—not a container—you can't change a portion of it:

```
put myHandler before the script of button 1 -- WON'T WORK
put "doThis" into line 2 of the script of me -- WON'T WORK
```

Instead, to change a part of a script, place the script in a variable, make the changes you need, then set the object's script to the variable:

```
put the script of thisCard into tempVar
put "--" before line 4 of tempVar
set the script of thisCard to tempVar
```

How to change a window's title

See Also:

How to remove a window's title bar, Object menu > Stack Inspector, Why is the card number in the window name?, Why is there an asterisk in the window name?, Recipe for finding out whether a window is open by title, label property, name property

You change the title that appears in a stack window's title bar by setting the stack's label property.

All windows in a Revolution application—editable window, modal dialog box, modeless dialog box, or palette—are stacks, and you can change the title of any of them with the label property. If the stack doesn't have a label, the stack's name is displayed in the title bar instead.

You can either set these properties in a handler or the message box, or use the Basic Properties pane in the stack's property inspector to specify a window title.

How to change an element in an array variable

See Also:

How to access a value in an array variable, How to create an array variable, How to find out whether a container is an array, How to store an array in a custom property, [] keyword

An array variable is a variable that contains a list of values. Each element of the list has a name, called the element's key. You specify an individual elements in an array by using the array's name, along with the key for the element in square brackets [].

You change an array variable's elements the same way you change any other variable, using the put command. For example, to put "Hello" into the element "myKey" of the array variable "myArray", use the following statement:

```
put "Hello" into myArray[myKey]
```

How to change an object's color

See Also:

How to change the color of text, How to find out the actual color of an object, How to make an object the same color as its owner, How to restore the default colors of an object, About colors and color references, Color Names Reference, Why don't buttons respect my color settings?, Recipe for setting the red channel of an object, backgroundColor property, borderColor property, bottomColor property, colorNames function, focusColor property, foregroundColor property, hiliteColor property, owner property, shadowColor property, topColor property

You change an object's color by setting the object's color properties. The most commonly properties are the foregroundColor (used for text and for lines in graphic objects) and backgroundColor (used to fill objects).

You can either set these properties in a handler, or use the Colors & Patterns pane of the object's property inspector to set all eight of an object's color properties.

Tip: If one of an object's color properties is set to empty, it inherits that color from the object that owns it. For example, if you set the foregroundColor of a card to "yellow", all that card's objects will have yellow text.

How to change an object's property profile

See Also:

How to change default settings for new objects, How to change the profile for all objects in a stack or card, About properties and property profiles, revProfile property

To switch an object's property profile, you set its revProfile property. You can either set this property in a handler or the message box, or use the object's property inspector.

To set an object's profile in the property inspector, click the arrow button at the upper right of the inspector palette. From the menu that appears, choose "Property Profile", then choose the profile name you want.

How to change behavior depending on the platform

See Also:

How to preview how a stack will look on other platforms, Supported Platforms Reference, lookAndFeel property, platform function

To find which platform a stack is running on, you use the platform function. You can then use an if/then control structure or a switch control structure to do different things depending on the platform.

```
on startup
  if the platform is "MacOS" then doMacStuff
  else if the platform is "Win32" then doWindowsStuff
  else doUnixStuff
end startup
```

How to change default settings for new objects

See Also:

How to change an object's property profile, How to create a card template for a stack, How to create a control, How to create a new card, How to create a stack, How to duplicate an object, Why do objects look different when created in a standalone?, Tip: Refer to a newly-created control, reset command, templateAudioClip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateImage keyword, templateGraphic keyword, templateGroup keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoClip keyword

To specify property settings for a newly created object, you set the properties of the template for the object's type before creating the object.

The following example creates several new scrolling fields whose text is in 9-point Geneva bold:

```
set the textFont of the templateField to "Geneva"
set the textSize of the templateField to 9
set the textStyle of the templateField to "bold"
set the vScrollbar of the templateField to true
repeat for 10 times
  create field
end repeat
```

How to change every chunk in a container

See Also:

How to shuffle the items or lines in a container, About chunk expressions, Recipe for building a repeated string, Recipe for finding lines that are not common to two containers, Recipe for reversing a string, Recipe for shifting the selection to the right or left, Recipe for word-wrapping text to a line length, character keyword, item keyword, line keyword, repeat control structure, word keyword

To make the same change to every line, item, word, or character in a container, you use a repeat control structure that iterates over each chunk in the container.

The following example places a colon (:) at the start of each line in a variable:

```
repeat with x = 1 to the number of lines of myVar
  put ":" before line x of myVar
end repeat
```

How to change the alignment of text

See Also:

How to change the font of text, How to change the size of text, How to change the style of text, Text menu > Align, `htmlText` property, `textAlign` property

You change the alignment of an object's text by setting the object's `textAlign` property. You can either set the `textAlign` in a handler or the message box, or select the object and choose Text menu>Align to change the object's alignment.

An object's text can be left-aligned, right-aligned, or centered:

```
set the textAlign of field "New" to "left"  
set the textAlign of the mouseControl to "right"  
set the textAlign of field ID 3580 to "center"
```

How to change the background color of text

See Also:

How to change the color of text, How to change the font of text, How to change the size of text, How to change the style of text, How to find out the actual color of an object, About colors and color references, Color Names Reference, Why do fonts and colors change when I create a standalone application?, Why don't buttons respect my color settings?, Recipe for a yellow highlight-marker effect, Text menu > Color, Shortcut to remove font changes from text, effective keyword, foregroundColor property, penColor property

You change the background color that text is drawn on by setting the text's backgroundColor property. You can either set the backgroundColor of a field (or chunk of a field) in a handler, or choose Text menu>Color to assign a color to the selected text or selected objects. You can also change the text color of objects using the Colors & Patterns pane of the object's property inspector.

Tip: A field's backgroundColor specifies the background color for all the text in the field, except for chunks of text for which you explicitly set a background color. In other words, the field's backgroundColor is the default font for all text in the field. To set the backgroundColor of all the text in the field to the field's color, use a statement like the following:

```
set the backgroundColor of char 1 to -1 of field 1 to empty
```

How to change the color of text

See Also:

How to change the background color of text, How to change the font of text, How to change the size of text, How to change the style of text, How to find out the actual color of an object, About colors and color references, Color Names Reference, Why do fonts and colors change when I create a standalone application?, Why don't buttons respect my color settings?, Recipe for handling selection of choices from a Text menu, Recipe for setting the red channel of an object, Text menu > Color, Shortcut to remove font changes from text, effective keyword, foregroundColor property, linkColor property, penColor property

You change the color of text by setting the text's foregroundColor property. You can either set the foregroundColor of a field (or chunk of a field) in a handler, or choose Text menu>Color to assign a color to the selected text or selected objects.

You can also change the text color of an object using the Colors & Patterns pane of the object's property inspector.

Tip: The "Color" submenu offers a pre-selected set of colors. To choose another color, choose Tools menu>Paint Tools. Click the box labeled "Pen Color" under the color wheel and choose the color you want. Choosing Text menu>Color>Pen Color now changes text to the color you chose.

How to change the current folder used for file operations

See Also:

How to find the Preferences folder, How to get the location of the user's home directory, How to list the contents of a folder, About filename specifications and file paths, Why can't Revolution find a file I specified?, defaultFolder property, files function, folders function

Certain file operations use the current folder. For example, the folders and files functions return all the folders or files in the current folder. Relative paths start from the current folder. And file dialog boxes display the current folder when they first appear.

You change the current folder using the defaultFolder property. By default, the current folder is the folder that contains the application, but you can change it to any folder using a statement like the following:

```
set the defaultFolder to "/Disk/Folder/Subfolder/"
```

Tip: To temporarily change the defaultFolder while your script performs an action, first save its value in a variable, then set it to the folder you want to use and perform the action, and finally set it to the contents of the variable.

How to change the cursor

See Also:

How to change tools, How to find out where the insertion point is, cursor property, lock cursor command, unlock cursor command

You change the shape of the cursor by setting the cursor property to the name of the cursor you want to display. The built-in cursors are arrow, hand, iBeam, watch, clock, busy, plus, cross, help, and none.

To use a custom cursor of your own design, set the cursor property to the ID of the image you want to use as a cursor.

Tip: The cursor automatically reverts to the standard cursor when the current handler finishes executing. To retain your specified cursor after the handler finishes, use the lock cursor command when you set the cursor property.

How to change the file type of an existing file

See Also:

How to assign creator and type signatures to a new file, How to rename a file, files function, fileType property

On Mac OS or OS X systems, each file incorporates a file type, which tell the operating system what kind of file it is. You use the do command with AppleScript to change the file type.

The following AppleScript statement changes the file type of a file called "My File" to "TEXT" (plain text file):

```
tell application "Finder" to set file type  
  of file "Disk:Folder:My File" to "TEXT"
```

You can place this AppleScript statement in a container (for example, a custom property called "ASTextChange"). Then use the following statement to execute the AppleScript code:

```
do the ATextChange of me as AppleScript
```

How to change the font of text

See Also:

How to change the alignment of text, How to change the color of text, How to change the size of text, How to change the style of text, Recipe for handling selection of choices from a Text menu, Text menu > Font, Shortcut to remove font changes from text, effective keyword, fontNames function, textFont property

You change the font of text by setting the text's textFont property. You can either set the textFont of a field (or chunk of a field) in a handler, or use the `Font` submenu in the Text menu to assign a font to the selected text or selected objects. You can also use the object's property inspector to change the text font.

A field's textFont specifies the font for all the text in the field, except for chunks of text for which you explicitly set a font. In other words, the field's textFont is the default font for all text in the field.

Tip: Choosing Text menu>Font>Use Owner's Font is equivalent to setting the text's textFont property to empty. This changes the selected text to the default font for the field it's in. If an object is selected, it changes the object's text to the default font of the card.

How to change the highlight state of a checkbox or button

See Also:

How to detect the highlight state of a control, How to find out the highlight state of a checkbox or button, How to give a checkbox a different state on each card, autoHilite property, hilite command, hilite property, radioButton property, unhilite command

The state of a control such as a radio button or checkbox is specified by the control's hilite property. To change the state, you set the control's hilite to true (for a selected radio button or checked box) or false (for an unselected radio button or unchecked box).

For radio buttons that are part of a group whose radioButton property is set to true, setting the hilite of one to true automatically sets the hilite of the rest to false.

You can set the hilite property for any button style, but other styles are not usually highlighted except when being pressed. (The button's autoHilite property does this automatically, so there's no need to set its hilite in a script.)

How to change the menu items in a button menu

See Also:

How to assign a keyboard equivalent to a menu item, How to change a menu's contents when it is opened, How to change the selected item in an option menu or combo box, How to create a separator line in a menu, Recipe for checkmarking a menu item, Object menu > Object Inspector, put command, text property

You change the menu items in a button menu by changing the contents of the button associated with the menu. Each line in the button's contents is used as a menu item in the menu.

You can either use the put command in a handler or the message box, or use the Basic Properties pane in the button's property inspector to change the button's contents. The following statement changes a button's contents to give it two menu items:

```
put "First" & return & "Last" into button "My Menu"
```

Tip: You can also set the button's text property: setting the text of a button to a string is equivalent to putting the string into the button.

How to change the order of cards in a stack

See Also:

How to bring a window to the front, How to change the tab order of controls on a card, How to go to another card, How to move one control in front of another, How to shuffle the cards in a stack, How to sort the contents of a field, Object menu > Card Inspector, View menu > Go First, View menu > Go Prev, View menu > Go Next, View menu > Go Last, Shortcut to go to the next card, Shortcut to go to the previous card, layer property, number property, sort command

The layer property of a card determines its order in the stack. For example, the first card's layer is 1. You set a card's position in a stack by setting the card's layer property to the position you want the card to be in.

You can either set this property in a handler or the message box, or use the Size & Position pane in the card's property inspector to change the card's layer.

When you change a card's layer, the card that was formerly in that position is shuffled back by one. For example, if you set a card's layer to 3, the third card becomes the fourth card in the stack.

How to change the pen color

See Also:

Color Names Reference, Text menu > Color > Pen Color, brushColor property, penColor property

The pen color is used by the Pencil, Line, and Curve paint tools, and is also a choice in the `Color` submenu of the Text menu. You change the pen color by changing the `penColor` property. You can either set this property in a handler, or use the Paint palette to select a pen color.

To select a pen color in the Paint palette, choose Tools menu>Paint Tools. Click the box labeled `Pen Color` under the color wheel and choose the color you want.

The following statement sets the `penColor` property to green:

```
set the penColor to "green"
```

How to change the profile for all the objects in a stack or card

See Also:

How to change an object's property profile, About properties and property profiles, revProfile property, revSetCardProfile command, revSetStackFileProfile command, revSetStackProfile command

You use the revSetCardProfile, revSetStackProfile, or revSetStackFileProfile command to change the property profiles of all objects in a card, stack, or stack file. The following example sets the profiles of all objects in a stack to "MyProfile":

```
revSetStackProfile "MyProfile","My Stack"
```

If any objects in the stack do not have a property profile called "MyProfile", this statement leaves them unaffected.

Tip: To automatically set all the objects in an application to use the same profile when you build the application, use the Profiles tab in Step 3 of the Distribution Builder.

How to change the selected button in a radio button cluster

See Also:

How to change the highlight state of a checkbox or button, How to create a radio button cluster, How to detect the highlight state of a control, `hilitedButton` property, `hilitedButtonID` property, `hilitedButtonName` property, `ID` property, `name` property, `number` property, `radioBehavior` property

A radio button cluster is a group made up of radio buttons. To implement the radio-group behavior (only one button highlighted at a time), the group's `radioBehavior` property is set to `true`.

You change the current setting of a radio button cluster by setting the group's `hilitedButton`, `hilitedButtonName`, or `hilitedButtonID` property to the number, name, or ID of the radio button you want to select. The following example selects a button named `Second` (where the radio button cluster might consist of `First`, `Second`, and `Third`):

```
set the hilitedButtonName of group "Channel" to "Second"
```

Tip: To unhighlight all the buttons in a radio button cluster, set the group's `hilitedButton` property to zero.

How to change the selected item in an option menu or combo box

See Also:

How to change a menu's contents when it is opened, How to change the menu items in a button menu, How to determine the current selection in a menu, label property, menuHistory property, menuPick message

Option menus and combo boxes have a current setting, which is visible in the menu even when the menu isn't open. You change this by setting the button's menuHistory property to the line number of the menu item. The following example chooses the second menu item in a menu:

```
set the menuHistory of button "Color" to 2
```

Note: Setting the menuHistory sends a menuPick message to the button.

Tip: If you know the text of the desired menu item, but not its line number, use the lineOffset function:

```
set the menuHistory of button "Color" to
```

```
lineOffset("Blue",button "Color")
```

How to change the size and position of a window

See Also:

How to center a window on the screen, How to change the size and position of an object, How to close a window, How to put different-sized cards in the same stack, How to prevent the user from resizing a window, How to respond to moving a window, How to respond to resizing a window, Object menu > Stack Inspector, formattedHeight property, formattedWidth property, height property, location property, rectangle property, revChangeWindowSize command, width property

You change a stack window's size and position by setting its rectangle and location properties. You can either set these properties in a handler, or use the stack's property inspector to change the stack's size and position.

In the Size & Position pane of the property inspector, use the Width and Height fields to change the stack's width and height.

Stacks have related properties such as left, bottom, topRight, and so on. You can change the location of a stack window's edges or corners by setting these properties in a handler.

How to change the size and position of an object

See Also:

How to change the size and position of a window, How to put different-sized cards in the same stack, Object menu > Align Selected Controls, Shortcut to equalize heights of selected controls, Shortcut to equalize height and width of a control, Shortcut to equalize widths of selected controls, height property, location property, rectangle property, width property

You change an object's size and position by setting its rectangle and location properties. You can either set these properties in a handler, or use the Size & Position pane in the object's property inspector to change the object's size and position.

In the property inspector, use the Width and Height fields to change the object's width and height. You can also adjust the object's location and rectangle.

Objects have related properties such as left, bottom, topRight, and so on. You can change the location of an object's edges or corners by setting these properties in a handler.

How to change the size and position of the video grabber window

See Also:

How to capture video from a video camera, How to change video grabber settings, How to take a picture with a video camera, `revInitializeVideoGrabber` command, `revSetVideoGrabberRect` command

You move and resize the video grabber window by using the `revSetVideoGrabberRect` command.

When you open the video grabber using the `revInitializeVideoGrabber` command, you specify a rectangle for the video grabber window. When you use the `revSetVideoGrabberRect` command, the video grabber window changes position (and size, if necessary) to fit in the rectangle you specify.

Note: Because the video grabber is not a stack window, it does not have properties, so you cannot set its rectangle, location, height, and width directly.

How to change the size of text

See Also:

How to change the alignment of text, How to change the color of text, How to change the font of text, How to change the style of text, Why do lines of text overlap?, Recipe for handling selection of choices from a Text menu, Text menu > Size, Shortcut to remove font changes from text, effective keyword, fontSizes function, formattedHeight property, formattedWidth property, lineHeight property, textSize property

You change the size of text by setting the text's textSize property. You can either set the textSize of a field or other object (or chunk of a field) in a handler, or use the "Size" submenu in the Text menu to assign a font to the selected text or selected objects.

You can also use the object's property inspector to change the text size.

Tip: Choosing Text menu>Font>Use Owner's Size is equivalent to setting the text's textSize property to empty. This changes the selected text to the default size for the field it's in.

How to change the style of text

See Also:

How to change the alignment of text, How to change the color of text, How to change the font of text, How to change the size of text, How to make subscripts and superscripts, How to remove a single style from text, How to remove all styles from text, Recipe for handling selection of choices from a Text menu, Shortcut to remove font changes from text, effective keyword, textStyle property

You change the style of text by setting the text's textStyle property. You can either set the textStyle of a field (or chunk of a field) in a handler, or use the Text menu to assign a style to the selected text or selected objects.

You can also use the object's property inspector to change the text style.

Tip: To add a style to existing styles in a handler (for example, to take bold text and make it bold and italic), use a statement like the following:

```
set the textStyle of line 2 of field "Example" to  
    (the textStyle of line 2 of field "Example", "italic")
```

How to change the tab order of controls on a card

See Also:

How to change the order of cards in a stack, How to move one control in front of another, Why doesn't the Tab key move to the next field?, Why is there a border around controls?, layer property, number property

You change the tab order of controls on a card by setting each control's layer property. The control with the lowest layer is first in the tab order. When you press the Tab key, the next control becomes active.

You can either set this property in a handler, or use the Align Objects pane of the property inspector. Select all the controls you want to relayer in the desired order and choose Object menu>Object Inspector, then choose "Align Objects" from the menu at the top of the inspector palette. Click the "Relayer Down" button at the bottom of the pane.

To re-layer the controls in a group, select the group and choose Object menu>Edit Group, then select the controls.

Tip: To remove a control from the tab order, set its traversalOn property to false.

How to change tools

See Also:

How to change the cursor, Tools menu > Browse Tool, Tools menu > Pointer Tool, Tools menu > Tools Palette, Shortcut to switch between browse and pointer tool, choose command, tool property

You change tools by using the choose command.

The current tool controls what happens when you click in an editable window. Use the Browse tool to interact with your stack, click buttons, type into fields, and so on. Use the Pointer tool to select and move objects. Use the other tools to create objects or to paint in image objects.

You can either use the choose command in a handler or the message box, or click the tool you want to use in the Tools palette or Paint palette.

Tip: You can find out which tool is active by checking the tool property in a handler.

How to change video grabber settings

See Also:

How to change the size and position of the video grabber window, `revSetVideoGrabSettings` command, `revVideoGrabDialog` command, `revVideoGrabSettings` function

You change the settings used for video capture in the video dialog box, which is displayed with the `revVideoGrabDialog` command. The settings you choose in the dialog box affect the currently-open video grabber window.

To save the current settings for later use, use the `revVideoGrabSettings` function and store the returned value in a custom property, a file, or other container:

set the savedSettings of me to `revVideoGrabSettings()`

You can then restore those settings in another session, using the `revSetVideoGrabSettings` command:

`revSetVideoGrabSettings` the savedSettings of me

How to check whether AppleScript is installed

See Also:

alternateLanguages function, do command

If your application uses the do command to execute AppleScript or another OSA language, you should check whether the language is installed before trying to execute code in it.

You check whether an OSA language is installed using the alternateLanguages function. The following example checks whether AppleScript is installed:

```
if "AppleScript" is not among  
    the lines of the alternateLanguages then  
    answer "AppleScript must be installed."  
    exit to top  
end if
```

How to clear the clipboard

See Also:

How to copy an image to the clipboard, Why is the selected text deselected?, clipboard property, clipboardData property, copy command, cut command, paste command

You clear the clipboard by setting the clipboardData property to empty:

set the clipboardData to empty

How to close a modal dialog box

See Also:

How to create a custom dialog box, How to display a dialog box, How to edit a palette or dialog box, How to make the Return or Enter key activate a button, About windows, palettes, and dialog boxes, Why does a stack window open in the wrong mode?, Shortcut to display a contextual menu, Tip: Trapped in a dialog box?, close command, modal command

When you create a modal dialog box, you should include at least one button with the following statement:

```
close this stack
```

Closing the stack that is being displayed as a modal dialog box dismisses the dialog box and allows the user to continue. Most modal dialog boxes have an OK button and Cancel button, and both buttons should include the statement above at the end of their mouseUp handlers.

Tip: You can change a modal dialog box to an editable window using the contextual menu shortcut Command-Shift-Control-click (Mac OS or OS X) or Control-Shift-Right-click (Unix or Windows). Choose "Top Level" from the "Stack Mode" submenu to change the dialog box to a normal editable window so you can close it.

How to close a window

See Also:

How to quit the application, How to remove a window's close box, How to respond to closing a window, How to show or hide a window, File menu > Close, File menu > Close and Remove From Memory..., close command, delete stack command, destroyStack property

You close a stack window by choosing File menu>Close or by using the close command in a handler or the message box. (All windows in a Revolution application—editable window, modal dialog box, modeless dialog box, or palette—are stacks, and any of them can be closed using the close command.)

Important! Closing a stack window does not purge the stack from memory if another stack in the same stack file is still open, or if the stack's destroyStack property is set to false. To close a stack and purge it from memory, either choose File menu>Close and Remove From Memory, or close all the stacks in the file and then use the delete stack command to purge the main stack from memory.

How to close the connection to a database

See Also:

How to save changes to a SQL database, About connecting to and using SQL databases, revCloseDatabase command, revOpenDatabases command

You use the revCloseDatabase command to close an open connection to a database.

When you open a database connection using the revOpenDatabase function, the function returns a database ID number. Since more than one database connection can be open at a time, you use this ID number to specify which connection you want to close:

```
put revOpenDatabase("MySQL",theHost,"RatesDB",,) into myDB
-- ... do database operations here ...
revCloseDatabase myDB -- use previously returned ID number
```

Note: When the application quits, all open database connections are closed automatically.

How to compare the contents of two containers

See Also:

= operator, <> operator, < operator, <= operator, > operator, >= operator, offset function, contains operator, is among operator

You use the =, <>, <=, >=, <, and > operators to compare two containers.

To test whether two containers contain the same thing, use the = and <> operators:

```
if myVariable = field "Test" then answer "Success!"  
if myVariable <> field "Test" then beep -- not equal
```

To compare a number in one container to another number, use the <=, >=, <, or > operator:

```
if it < 22 then answer "Value too small."
```

How to convert a MetaCard stack to Revolution

See Also:

About Revolution for MetaCard developers

MetaCard and Revolution use the same file format for stack files, so to open a MetaCard file in Revolution, you simply choose File menu>Open Stack and locate the MetaCard file you want to open.

How to convert a SuperCard project to Revolution

See Also:

How to convert a MetaCard stack to Revolution, How to import a HyperCard stack, About Revolution for SuperCard developers

To convert a SuperCard project to a Revolution stack, you download the converter from <http://www.runrev.com/revolution/sctorev/sctorev.sit>. The converter consists of a SuperCard project and a Revolution stack. Documentation is included with the converter program.

The process works like this:

1. Open the "SuperCard -> SIF Exporter" in SuperCard and export your project. The Exporter creates a set of text and image files.
2. Open the "SIF to Rev importer.rev" stack in Revolution and import your exported files. The Importer creates a Revolution stack file from the text and image files.

How to convert an image to a different format

See Also:

How to create a screenshot of a stack, How to display a TIFF file, How to import a picture file into an existing image object, Why can't I export an image to a GIF file?, Shortcut to make an image transparent, export command, import command, jpegQuality property, paintCompression property

Images are stored in one of the formats supported by Revolution: GIF, JPEG, PNG, PICT (on Mac OS and OS X only), BMP, PBM/PGM/PPM, and XBM/XPM/XWD formats, as well as its own internal format. To change the format of an image, do the following:

1. Set the global paintCompression property to the desired format.
2. Use any of the paint tools to make a change to the image. (Even a small change, such as changing the color of a single pixel and then changing it back, is enough.)
3. Close and re-open the stack. The image is converted to the new format.

How to convert between Unicode (UTF-16) and UTF-8 text

See Also:

How to convert between Unicode and ASCII text, How to enter or display Unicode text in a field, How to find out whether text in a field is Unicode, How to import a Unicode text file, How to import Shift-JIS Japanese text, uniDecode function, uniEncode function

Revolution displays non-Roman-alphabet languages using Unicode (UTF-16). You use the uniDecode and uniEncode functions to convert between UTF-16 and UTF-8.

The following statement converts a variable's contents from UTF-8 to UTF-16, and places the resulting Unicode text in a field:

```
put uniEncode(myVariable,"UTF8") into field "My Field"
```

How to convert between Unicode and ASCII text

See Also:

How to enter or display Unicode text in a field, How to find out whether text in a field is Unicode, How to import a Unicode text file, uniEncode function, uniDecode function, useUnicode property

You use the uniEncode and uniDecode functions to convert text from double-byte (Unicode) to single-byte ASCII, or vice versa.

To convert a string of single-byte characters to Unicode text, use a statement like the following:

```
put uniEncode(field "Text") into myUnicodeText
```

To convert a string of double-byte characters to single-byte, use a statement like the following:

```
put uniDecode(the unicodeText of field "Japanese Text")  
into convertedText
```

How to convert line endings when transferring a file

See Also:

How to download a binary file, About URLs, uploading, and downloading, Why is there a problem with line endings?, Why was a downloaded file corrupted?, file keyword, URL keyword

Different operating systems use different characters to mark the end of a line. Mac OS and OS X use a return character (ASCII 13), Unix systems use a linefeed character (ASCII 10), and Windows systems use a return followed by a linefeed. To avoid problems when transporting a stack between platforms, Revolution always uses linefeeds internally.

When you use a file URL as a container. Revolution translates as needed between the your system's end-of-line marker and Revolution's linefeed character. The following example downloads a text file from an FTP server, converting its end-of-line markers as needed:

```
put URL "ftp://ftp.example.org/myfile.txt"
```

```
into URL "file:myfile.txt"
```

How to copy a card

See Also:

How to move a card within a stack, How to move a card between stacks, copy command, paste command

You copy a card by selecting it and choosing Edit menu>Copy, or by using the copy command in a handler or the message box.

To select a card, double-click it with the Pointer tool. The selected card can now be copied and pasted.

The following statement copies the current card to the clipboard:

```
copy this card
```

After copying, go to where you want to place the card in, then either choose Edit menu>Paste or use the paste command. The card is pasted into the current stack, after the card you were on.

How to copy a resource fork

See Also:

About file types, application signatures, and file ownership, About using URLs, uploading, and downloading, copyResource function, deleteResource function, getResource function, getResources function, resfile keyword, setResource function

On Mac OS and OS X systems, files consist of either or both of a data fork and a resource fork. The resource fork contains defined resources such as icons, menu definitions, dialog boxes, fonts, and so forth. To copy a resource fork from one file to another, you use the resfile URL scheme:

```
put URL "resfile:/Disk/Folder/source.rsrc"
```

```
into URL "resfile:/Disk/Apps/destFile"
```

Important! If the destination file doesn't exist, you must create it before copying the resource fork to it:

```
put empty into URL "file:/Disk/Apps/destFile"
```

Tip: To copy a single resource, use the copyResource function.

How to copy an image to the clipboard

See Also:

How to display the paint tools, How to duplicate an object, Edit menu > Copy, clipboardData property, copy command, cut command, paste command

To manually copy the contents of an image, so that it can be pasted into another image or in another application, you either set the clipboardData property or use the Select tool.

To copy the image contents manually, first choose the Select tool from the Paint Tools palette. Drag to select the contents of the image, then choose Edit menu>Copy.

To copy the contents of an image in a handler or the message box, use a statement like the following:

set the clipboardData["image"] to image "Bulb"

How to copy dragged and dropped text

See Also:

How to allow copying text from a locked field, How to copy text between fields, How to prevent dragging and dropping to a field, How to respond to a drag and drop, dragEnd message

Normally, when you drag and drop a text selection between fields, the text is moved from the source field to the destination field. This means that the text is removed from the dragSource and placed in the dragDestination.

This automatic behavior is triggered when Revolution receives the dragDrop and dragEnd messages. Specifically, the dragged text is removed from the source field when Revolution receives the dragEnd message. Trapping the message and not allowing it to pass prevents Revolution from removing the text:

```
on dragEnd -- in source field script
  -- do nothing - just trap the message without passing
end dragEnd
```

How to copy styled text between fields

See Also:

How to copy an image to the clipboard, How to copy text between fields, How to import and export styled text, How to store styled text in a variable or property, Recipe for collecting text selections on the clipboard, Recipe for removing boldfacing from a field, `htmlText` property

You copy styled text—that is, text that includes font, size, style, and color information—between fields using the `htmlText` property of the fields.

The following statement copies the text in the field `Source` to the field `Destination`, preserving all style information:

```
set the htmlText of field "Destination" to
```

```
the htmlText of field "Source"
```

Tip: To copy only the text itself, without the style information, use the `put` command.

How to copy text between fields

See Also:

How to allow copying text from a locked field, How to copy an image to the clipboard, How to copy dragged and dropped text, How to copy styled text between fields, How to display the same text on multiple cards, Recipe for collecting text selections on the clipboard, File menu > Copy, File menu > Paste, put command, text property

You copy text between fields by choosing Edit menu>Copy or by using the put command in a handler.

After you use the ôCopy Textö menu item, click where you want to paste the text, then choose Edit menu>Paste.

The following statement copies the contents of the field ôSourceö to the field öDestinationö:

```
put field "Source" into field "Destination"
```

Tip: Use the before and after keywords with the put command to add new text without deleting text already in the destination field.

How to create a card template for a stack

See Also:

How to automatically include groups on a new card, How to change default settings for new objects, About groups and backgrounds, `backgroundBehavior` property, `create` command, `templateCard` keyword

You may want to create many cards with the same layout. One way to make this process easier is to create a template card, then create all new cards from the template.

To create a card that will serve as a template for other cards you create in a stack, you group the objects on the template card, then set the `backgroundBehavior` property of the group (or groups) to `true`. You set the `backgroundBehavior` property in a handler or on the Basic Properties pane of the group's property inspector.

When you create a new card, any groups whose `backgroundBehavior` is `true` that are on the current card are automatically placed on the new card.

How to create a code library

See Also:

How to add custom utilities to the development environment, How to execute Transcript code in a file, How to include an object in the message path of every object, How to intercept a message, How to install a plugin, How to use a handler outside the message path, About commands and functions, About messages and the message path, Why does an unwanted handler run?, Why isn't a handler executed?, Development menu > Object Library, backScripts function, frontScripts function, insert script command, start using command

A library is a set of custom commands and custom functions for a specific application or a specific area of functionality. For example, if you're writing an image-transformation application, you might want to collect your transformation handlers in one place, where you can easily use any of them from any script in your application.

To create a code library, place the handlers you want to re-use in any object that's available in your stack, then use the insert script command to add that object to the message path, as in the following example:

```
insert script of group "MyLib" into back
```

Handlers in that object's script are now accessible to any other handler in Revolution.

How to create a control

See Also:

How to change default settings for new objects, How to draw a shape, Why is a control the wrong size when created by a handler?, Tools menu > Tools Palette, Object menu > New Control, copy command, create command

You create a control by choosing the type of control you want from Object menu>New Control, or by choosing a tool (other than Browse or Pointer) from the Tools palette and clicking or dragging in a stack window, or by using the create command in a handler or the message box.

The items in the "New Control" submenu create the control you choose in the current stack, using the default sizes in the "Sizes and Appearance" pane of the Preferences window.

The create command creates the new control you specify.

Tip: After creating a new control, the Pointer tool is chosen automatically. To switch back to the Browse tool, use the choose command.

How to create a custom dialog box

See Also:

How to close a modal dialog box, How to display a dialog box, How to edit a palette or dialog box, How to make the Return or Enter key activate a button, About windows, palettes, and dialog boxes, Why does a stack window open in the wrong mode?, Recipe for speaking an alert message, Shortcut to display a contextual menu, answer command, ask command, dialogData property, modal command, modeless command, style property, topLevel command

You create a custom dialog box by creating a stack that includes all the buttons, text fields, and other controls you need in the dialog box. Then use the modal command to display the stack as a modal dialog box. To display a stack as a modeless dialog box, use the modeless command instead.

It is usually simplest to design dialog boxes as substacks of your application's main stack. For example, if you have a substack called "Select Options", you can display it as a dialog box using the following statement:

```
modal "Select Options"
```

Note: You can instead set the stack's style property to "modal" to force it to be displayed as a modal dialog box. However, this is less versatile, since it means that if you want to edit the stack, you have to set the stack's style to "topLevel" instead of simply opening it.

How to create a custom printed report

See Also:

How to display the print settings dialog box, How to print a card, How to print all the cards in a stack, How to print a report, How to print in landscape mode, How to print selected cards as a single print job, Recipe for printing the fields on a card, File menu > Print Card..., print command

You create a custom report by creating the layout you need in a stack window (using fields, graphics, and other controls as needed to create your report template); using the put command and htmlText properties to copy unformatted or formatted text from another stack to the fields in the report stack; then using the print card command to print the report.

This basic technique can be used to create any sort of report. To make the report template stack invisible to the user during printing, set its visible property to false. To print multiple pages, use the open printing command to start a batch print job, then use the print card command for each layout and data set you want to print. You can either create the report template stack with a handler, or create it during development and save it as a substack of your main stack.

Tip: To easily create a report without scripting, use the Report Builder.

How to create a custom property set

See Also:

How to create a custom property, How to delete a custom property set, How to duplicate a custom property set, How to refer to a custom property in a non-active set, How to rename a custom property set, About custom properties and custom property sets, customPropertySet property, customPropertySets property

You create a custom property set of an object by setting its customPropertySet property to the name of the new set. If the custom property set does not already exist, Revolution creates it:

```
set the customPropertySet of field "Hello" to "NewPropSet"  
-- creates newPropSet if it doesn't already exist  
-- also makes newPropSet the current property set  
-- now we'll create a new custom property in it:  
set the myNewProp of button "Hello" to false
```

You can either create a new custom property set for an object in a handler or the message box, or use the Custom Properties pane in the object's property inspector to create the custom property set.

How to create a custom property

See Also:

How to create a custom property set, How to delete a custom property, How to find out whether a custom property exists, How to rename a custom property, How to store an array in a custom property, About custom properties and custom property sets, Why doesn't a custom property appear in the property inspector?, customKeys property, customProperties property

You create a custom property of an object by using the set command to put something into the new custom property. If the custom property does not already exist, Revolution creates it:

```
set the newProperty of this stack to true
-- creates newProperty if it doesn't already exist
```

You can either set the new custom property in a handler or the message box, or use the Custom Properties pane in the object's property inspector to create the custom property.

Note: You can create custom properties for any object, but you cannot create global custom properties.

How to create a directory on an FTP server

See Also:

How to use a password for an FTP server, About using URLs, uploading, and downloading, create folder command, ftp keyword, libURLftpCommand function, libURLftpUpload command

You create a directory on an FTP server by uploading a file to the new (nonexistent) directory. If you attempt to upload a file to a nonexistent directory, the directory is automatically created. You can then delete the file, if you wish, leaving a new, empty directory on the server:

```
-- Create an empty file in the nonexistent directory:  
put empty into
```

```
URL "ftp://name:pass@example.com/newdir/dummy"
```

```
-- Delete unwanted empty file to leave new directory:  
delete URL "ftp://name:pass@example.com/newdir/dummy"
```

Note: This method can be used to create only one level of directory: the parent directory of the new directory must already exist on the server.

How to create a file

See Also:

How to assign creator and type signatures to a new file, How to associate files with a Mac OS standalone application, How to associate files with an OS X standalone application, How to associate files with a Windows standalone application, How to delete a file, How to determine whether a file exists, How to export data to a file, How to make an alias, symbolic link, or shortcut to a file, How to rename a file, How to store preferences or data for a standalone application, About using URLs, uploading, and downloading, File menu > Save As..., binfile keyword, export command, file keyword, put command, there is a operator, there is no operator, URL keyword

You create a file by putting something into a file or binfile URL. Suppose you have a variable called `myName` that contains the desired file path. The following statement creates a file with the specified name and location that contains the contents of a field:

```
put field 1 into URL ("file:" & myName)
```

If you put empty into the URL of a nonexistent file, the file is created as an empty file. For example, the following statement creates an empty file called "New File.txt" in the current folder:

```
put empty into URL "file:New File.txt"
```

How to create a hypertext link

See Also:

Why is some text blue and underlined?, Text menu > Link, clickChunk function, clickText function, link keyword, hide groups command, show groups command, textStyle property

To create a hypertext link in a field, you use the clickText function in a mouseDown or mouseUp handler to obtain the text the user clicked, then do something with that text. This example goes to the card whose name was clicked:

```
on mouseDown
  if there is a card (the clickText)
    then go card (the clickText)
end mouseDown
```

Tip: To make multi-word phrases into links, set the textStyle of the phrase to `linkö`.

How to create a new card

See Also:

How to change default settings for new objects, How to create a stack, Why do unwanted objects appear on new cards?, Object menu > New Card, create card command, templateCard keyword

You create a card by choosing Object menu>New Card, or by using the create card command in a handler or in the message box. The new card is created in the current stack, after the current card.

Tip: When creating several cards with the same properties, set the properties of the templateCard before creating your cards. Newly created cards are automatically given the property settings of the templateCard, so setting up the template first lets you save time.

How to create a palette

See Also:

How to edit a palette or dialog box, activatePalettes property, hidePalettes property, palette command, raisePalettes property, style property

You create a palette by creating a stack with whatever buttons (or other controls) you want in the palette.

Then use the palette command to display the stack in a palette window:

```
palette stack "My Palette"
```

How to create a radio button cluster

See Also:

How to change the selected button in a radio button cluster, How to give a border to a radio button cluster, Object menu > Group, Object menu > New Control > Radio Button, autoHilite property, hilitedButtonName property, radioBehavior property

You create a radio button cluster by creating several radio buttons, then grouping them. Once you group them, the radio buttons automatically adopt the behavior of a radio button cluster: only one button can be highlighted at a time, and clicking one of the buttons automatically unhighlights the rest of the buttons in the group.

To group a set of radio buttons into a single radio cluster, select all the buttons, then choose Object menu>Group. The set of buttons becomes a single group, where only one button in the group can be highlighted at a time.

How to create a record set in a SQL database

See Also:

How to find out which record sets are open in a database, How to move through the records in a record set (database cursor), About connecting to and using SQL databases, revCloseCursor command, revOpenDatabase function, revQueryDatabase function, revQueryDatabaseBLOB function

You use the revQueryDatabase or revQueryDatabaseBLOB function to create a record set (database cursor) in a SQL database.

In the revQueryDatabase or revQueryDatabaseBLOB function, you provide a SQL SELECT statement that describes records you want to select. (For example, you might want to work with only records where the STATE field is 'CA'.)

The set of selected records is called a record set. You can refer to the record set using the ID number returned by the function.

How to create a report layout card

See Also:

How to create and store report printing settings for later use, How to display the contents of a field in a printed report, How to preview a printed report, How to print a report, How to specify what cards to include in a printed report, Tools menu > Report Builder, Object menu > New Control > Report Object, revPrintReport command

A report layout card is a card that includes one or more report viewer objects, and may include other objects (such as graphics or images). Each report viewer is linked to a field (which can be in any stack).

To create a report layout card, first create a new card (which can reside in any stack). Create a report viewer object for each field that you want to display in the printed report, and use the report viewer's property inspector to link it to the field.

If you add other controls to the report layout card, they will appear on each card of the printed report. For example, you can add line graphics to make vertical or horizontal rules, an image containing a company logo, or other objects to create whatever layout design you want.

How to create a screenshot of a stack

See Also:

File menu > Import As Control > Snapshot, export command, import snapshot command

You take screenshots by choosing File menu>Import As Control>Snapshot, or by using the import snapshot command in a handler or the message box. The following statement takes a screenshot of the stack `My Stack` and makes it into an image into the current stack:

```
import snapshot from rect (the rect of stack "My Stack")
```

Tip: To create a file for the screenshot instead of importing it as an image object, first use the import snapshot command, then export the image to a file, and finally delete the image:

```
import snapshot from rect (the rect of stack "My Stack")  
export last image to file "screenshot.jpg" as PNG  
delete last image -- clean up
```

How to create a scrolling window

See Also:

hScroll property, hScrollbar property, lockLocation property, vScroll property, vScrollbar property

You create a scrolling window by placing all the objects on the card in a scrolling group. If you make the group the same size as the stack window, scrolling the group causes the whole content area of the window to scroll.

To display the group's horizontal or vertical scroll bar, set the group's hScrollbar or vScrollbar property to true. You can either set these properties in a handler or the message box, or use the Basic Properties pane in the group's property inspector to display or hide a vertical or horizontal scroll bar.

Tip: After sizing the group to fit the window, set the group's lockLocation property to true. This prevents it from automatically resizing to fit the contents when you switch cards.

How to create a separator line in a menu

See Also:

How to change the menu items in a button menu, Menu Builder Tutorial, Object menu > Object Inspector

To create a separator line in a menu button, you place a dash (-) on a separate line in the menu contents. When the menu is drawn, the dash is rendered as a separator line.

You can add a dash to a menu by placing it in the contents of the button that the menu is associated with, either in the Basic Properties pane of the property inspector or using the set command to set the button's text property.

If the menu is in a menu bar, you can use the Separator button (to the left of the Delete Item button) in the Menu Builder to add a separator line below the currently selected menu item.

How to create a spreadsheet-style table

See Also:

How to allow editing of individual cells in a table field, How to format numbers in a table field

A table field is a field that is displayed as a grid, in the style of a spreadsheet. Each line of the field becomes a row, and the columns in a row are separated by tab characters. You control a table field's behavior using the Table pane of the field's property inspector.

To make a field into a table field, follow these steps:

1. Open the field's property inspector and choose "Table" from the menu at the top of the inspector palette.
2. Check the "Table object" box to make the field into a table.
3. If you want, change the baselines, grid, and tab stops settings to change the appearance of the table field.

How to create a stack

See Also:

How to change default settings for new objects, How to create a card, File menu > New Mainstack, File menu > New Substack of (mainstack name), create stack command, templateStack keyword

You create a new stack by choosing File menu>New Mainstack or File menu>New Substack, or by using the create stack command in a handler or the message box.

The `new Mainstack` menu item creates a new main stack, which will be the first stack in its stack file. Use it if you're starting work on a new application or creating a new file. The `new Substack` menu item creates a substack that will be included in the current stack's stack file. Use it if you're creating a subsidiary window for the application you're working on.

The create stack command creates a new main stack. To make the new stack a substack of an open main stack, set its `mainStack` property to the name of the stack.

How to create a standalone application

See Also:

How to assign a custom creator signature to a standalone, How to distribute a stack to users on other platforms, How to save and re-use Distribution Builder configurations, Why can't I create a standalone MacOS application?, Supported Platforms Reference, File menu > Build Distribution..., Getting Started Tutorial, environment function

You create a standalone application using the Distribution Builder. To open the Distribution Builder, choose File menu>Build Distribution.

Use the Platform checkboxes in the Distribution Builder to indicate which platforms you want to build for. Since programs for one platform cannot run on others, the Distribution Builder creates a separate folder for each platform.

Important! You cannot build Mac OS applications on Unix or Windows systems. This is because these platforms require a file format that's not supported on Unix and Windows.

How to create a synonym for a command or function

See Also:

How to create a code library, How to find out whether a command or function is defined, About commands and functions

At times, you may want to be able to call a command or function by another name. For example, you might want to create a shorter synonym for a command so it's easier to type in the message box.

You create a synonym for any command or function by creating a handler that calls the command or function. For example, if the following handler is in the message path, you can use the `libURLDownloadToFile` command with the shorter synonym `dl`:

```
on dl theURL,theFilePath
    libURLDownloadToFile theURL,theFilePath
end dl
```

To create a synonym for a function, you use a similar method, but with a function handler instead of a message handler.

How to create a window the same size as the screen

See Also:

backdrop property, rectangle property, screenRect function, windowBoundingRect property

To create a window the same size as the screen, you set the rectangle of the window to the screenRect, using a statement such as the following:

set the rect of stack "Big Stack" to the screenRect

The screenRect function returns the area of the main screen.

Tip: To place a colored or patterned backdrop behind the application without changing the window size, use the backdrop property as in the following statement:

set the backdrop to "black"

How to create an array variable

See Also:

How to access a value in an array variable, How to declare a variable, How to find out whether a container is an array, How to return an array from a function, How to store an array in a custom property, [] keyword, keys function

You create an array variable the same way you create an ordinary variable: by either declaring it with the local or global command, or by putting something into it with the put command.

Similarly, you create a new element within an existing array variable by putting something into one of its elements. You specify an element by placing the element's key in square brackets after the variable's name. For example, to put "Hello World" into an element named "message" in the variable "myArray", use the following statement:

```
put "Hello World" into myArray["message"]
```

If the element "message" doesn't already exist in the variable, this statement creates it.

How to create and store report printing settings for later use

See Also:

How to create a report layout card, How to preview a printed report, How to print a report, How to specify what cards to include in a printed report, Tools menu > Report Builder

You use the Report Builder to create settings for printed reports. (You can either print the report from the Report Builder window, or use the `revPrintReport` command.) To open the Report Builder, choose Tools menu>Report Builder.

In the Report Builder window, you can choose between printing one card per page or as many cards as will fit, set page margins and space between cards, specify which cards you want to include in the report, and choose a stack to use as a header and/or footer for the report.

When you close the Report Builder window, the settings are stored on the current card. The next time you print a report while on that card, the same settings will be used.

How to create and use a numbered set of variables

See Also:

How to create an array variable, How to declare a variable, How to delete a variable, About containers, variables, and sources of value

To easily create a set of variables with similar names (such as `myVar1`, `myVar2`, ..., `myVar20`), you can use the concatenation operator `&`.

The following example shows how to use a set of variables named `myVar1`, `myVar2`, ..., `myVar20`:

```
repeat with x = 1 to 20
  put empty into ("myVar" & x)
end repeat
```

How to declare a variable

See Also:

How to create an array variable, Why does a variable lose its value?, constant command, explicitVariables property, delete variable command, global command, local command, variableNames function

Local variables you use within a handler do not need to be declared; they are created automatically when you use the put command to put something in them. Optionally, you can use the local command to declare a local variable.

Each global variable must be declared in every handler that uses it. You use the global command to declare a global variable. Variables that are shared by all the handlers in a script must be declared in the script, outside any handler, with the local command.

Tip: To require declaring all variables (including local variables), set the explicitVariables property to true.

How to delay an action

See Also:

How to cancel a pending message, How to schedule a future message, send command

You use the send in time form of the send command to delay an action.

When you specify a delay time with the send command, the message you send is queued until the specified time has passed. The following statement beeps twice after 20 seconds:

```
send "beep" to me in 20 seconds
```

Important! This form of the send command does not guarantee message delivery. If another handler is running when the specified time elapses, that handler must complete execution before the queued message is sent. The specified time is a minimum time.

How to delete a custom property set

See Also:

How to create a custom property set, How to delete a custom property, How to delete all an object's custom properties, How to duplicate a custom property set, How to rename a custom property set, About custom properties and custom property sets, customProperties property, customPropertySets property

You delete a custom property set by selecting it and clicking the Delete button in the Custom Properties pane of the property inspector, or by writing a handler.

In Transcript, you can't directly delete a custom property set. Instead, you place all the custom property set names in a variable, delete the one you don't want from that variable, and set the customPropertySets back to the modified contents of the variable. For example, the following statements delete a custom property set called "mySet" from the button "My Button":

```
get the customPropertySets of button "My Button"  
set the wholeMatches to true  
delete line lineOffset("mySet",it) of it  
set the customPropertySets of button "My Button" to it
```


How to delete a custom property

See Also:

How to create a custom property, How to delete all an object's custom properties, How to rename a custom property, About custom properties and custom property sets, Why doesn't a custom property appear in the property inspector?, customKeys property, customProperties property, wholeMatches property

You delete a custom property by selecting it and clicking the Delete () button in the Custom Properties pane in the property inspector, or by writing a handler.

In Transcript, you can't directly delete a custom property. Instead, you place all the custom property names in a variable, delete the one you don't want from that variable, and set the custom properties back to the modified contents of the variable. For example, the following statements delete a custom property called `propertyToRemove` from the button `My Button`:

```
get the customKeys of button "My Button"  
set the wholeMatches to true  
delete line lineOffset("propertyToRemove",it) of it  
set the customKeys of button "My Button" to it
```

How to delete a file

See Also:

How to create a file, How to determine whether a file exists, How to remove a file from an FTP server, How to rename a file, About filename specifications and file paths, delete file command, revDeleteFolder command

You use the delete file command to remove a file.

To delete a file, use the file's absolute path or relative path, as in the following statements:

```
delete file "/Volumes/External/Folder/File"  
delete file "../picture.jpg"  
delete file "mystuff.txt"
```

Be careful when deleting files. The delete file command does not place the file in the Trash or Recycle Bin; it removes the file completely from the disk, and cannot be undone.

Tip: To delete a folder (along with all its contents), use the revDeleteFolder command.

How to delete a file's resource fork

See Also:

How to copy a resource fork, How to delete a file, About file types, application signatures, and file ownership, About using URLs, uploading, and downloading, deleteResource function, delete URL command

On Mac OS and OS X systems, files consist of either or both of a data fork and a resource fork. The resource fork contains defined resources such as icons, menu definitions, dialog boxes, fonts, and so forth.

You use the delete URL command with a resfile URL to delete a resource fork.:

```
delete URL "resfile:/Disk/Folder/Picture.gif"
```

This statement deletes the file's resource fork and all the resources in it, but leaves its data fork intact.

Tip: To delete a single resource instead of the entire resource fork, use the deleteResource function.

How to delete a line from a container

See Also:

delete chunk command, line keyword

You use the delete chunk command to remove a line from a container (such as a variable or field). The following statement deletes the next-to-last line from a field:

```
delete line -2 of field "List"
```

To remove the contents of the line, but leave a blank line in the container, put empty into the line instead:

```
put empty into line 2 of myVar
```

How to delete a variable

See Also:

How to declare a variable, How to delete an element from an array variable, About containers, variables, and sources of value, Development menu > Variable Watcher, delete variable command, empty constant

You use the delete variable command to remove a variable from memory. The following example deletes a global variable:

```
delete global variable "myGlobal"
```

Tip: To delete a variable's contents without deleting the variable itself, put empty into the variable:

```
put empty into myVariable -- myVariable still exists
```

How to delete all an object's custom properties

See Also:

How to delete a custom property, How to delete a custom property set, How to list an object's custom properties, About custom properties and custom property sets, customProperties property, customPropertySets property

You can either delete an object's custom properties in a handler or the message box, or use the Custom Properties pane in the object's property inspector.

To delete all the custom properties in the current custom property set, set the customProperties property of the object to empty:

set the customProperties of the target to empty

To delete all the custom property sets (other than the default custom property set) of an object , along with all the properties they contain, set the customPropertySets property of the object to empty:

set the customPropertySets of the target to empty

How to delete an element from an array variable

See Also:

How to access a value in an array variable, How to create an array variable, How to delete a variable, How to find out whether a container is an array, About containers, variables, and sources of value, [] keyword, delete variable command, empty constant

You use the delete variable command to remove one element from an array variable. To delete an element, you specify both the variable and the element's key:

```
delete variable myVar["myElement"]
```

This statement removes the element named `myElement` from the variable `myVar`, but does not delete the variable itself.

Tip: To delete the contents of an element without deleting the element itself, put empty into the element:

```
put empty into myVar["myElement"]
```

How to delete an object

See Also:

How to select a group, How to make objects appear on more than one card, Edit menu > Clear, Object menu > Remove Group, delete command, remove command

You delete an object by selecting it and choosing Edit menu>Clear, or by using the delete command in a handler or the message box.

To remove a group from the current card without removing it from the stack, select the group and choose Object menu>Remove Group. In a handler or the message box, use the remove command.

How to dereference a variable

See Also:

About containers, variables, and sources of value, How to create a variable, @ keyword, do command, value function

To use the value of a variable instead of the variable's name in an expression, you use the `do` command.

The last line in the following example shows how to dereference a variable named `varName`, which contains the name of another variable, to obtain the value of the variable whose name is in `varName`:

```
put 1 into firstVariable
put "firstVariable" into varName
answer varName -- displays "firstVariable"
do "answer" && varName -- displays "1"
```

Tip: To use the contents of a variable in an expression (instead of its name), use the `value` function.

How to deselect all the entries in a list field

See Also:

How to deselect the selected text, How to determine which line of a list field is selected, How to select an entry in a list field, Why can't I leave all lines of a list field unselected?, hilitedLine property, listBehavior property

Normally, one or more lines in a list field are selected. You change the selected line or lines by setting the field's hilitedLine property.

To deselect all the entries, so that none of the lines in the list field is selected, set the field's hilitedLine property to empty:

set the hilitedLine of field "Records" to empty

Note: Changing the focus to a list field automatically selects the first line of the list, if none of the lines is currently selected.

How to deselect the selected text

See Also:

How to deselect all the entries in a list field, How to select text in a field, select command, selection keyword

You deselect the text selection using the following statement:

```
select empty
```

This statement deselects any selected text and removes the insertion point from any field itÆs in.

How to detect the highlight state of a control

See Also:

How to change the highlight state of a checkbox or button, How to give a checkbox a different state on each card, autoHilite property, hilite property, hilitedButton property

The state of a control such as a radio button or checkbox is specified by the control's hilite property. To find out whether the control is highlighted, you check the hilite property with a statement such as the following:

```
if the hilite of button "Status" is true then showStatus
```

Note: For radio buttons that are part of a group whose radioBehavior property is set to true, setting the hilite of one to true automatically sets the hilite of the rest to false. To find out which button in a group is highlighted, check the hilitedButton , hilitedButtonID, or hilitedButtonName property of the group:

```
if the hilitedButton of group "Color" is 1  
then set the backgroundColor of me to "blue"
```

How to determine the current selection in a menu

See Also:

How to change the menu items in a button menu, How to change the selected item in an option menu or combo box, How to simulate a menu choice, Recipe for checkmarking a menu item, menuHistory property, menuPick message, selectedText function

You find out which menu item is currently chosen in a menu (either an option menu or combo box menu) using the menu's selectedText function.

The following statement changes a property based on the current selection in a menu named `ôColorö`:

```
if the selectedText of button "Color" is "Light Red"
then set the backdrop to "#FF3333"
```

Tip: The selectedText function reports the actual text of the menu item. To find the number of the menu item, use the menuHistory property.

How to determine the dimensions of an image

See Also:

How to automatically adjust objects when a window is resized, Why does an object change size?, Recipe for getting the dimensions of a picture file, formattedHeight property, formattedWidth property, height property, width property

Usually, an image is displayed at its full size with no scaling. However, if the image's rectangle (or related properties such as its height and width) has been changed, the image may be shown larger or smaller than its native size. In this case, you may want to obtain the image's true dimensions.

You find out the true dimensions of an image using the formattedHeight and formattedWidth properties. The formattedWidth and formattedHeight, unlike the width and height, do not change when you resize the image object. The following statement resizes an image to its true width:

set the width of image "My Image" to

the formattedWidth of image "My Image"

How to determine the size of a file

See Also:

files function

You find out the size of a file using the detailed files form of the files function. The second item of the detailed files listing gives the size of the file. (On Mac OS and OS X systems, the size of the file's resource fork is the third item: this must be added to the second item to obtain the total file size.)

The following set of statements gets the size of a file called "MyFile.txt" in the defaultFolder and places it in a variable called myFileSize:

```
get the detailed files
filter it with "MyFile.txt,*"
put item 2 of it + item 3 of it into myFileSize
```

How to determine what kind of object triggered a message

See Also:

How to find out the object type of an object, me keyword, selectedObject function, target function

You determine which object was first sent a message using the target function. Since the target function returns the object's abbreviated name property, you can examine the first word of the return value to find out what kind of object the user interacted with to trigger the message.

The following statement, which might be found in a mouseUp handler in a card or stack script, finds out whether the message was sent by clicking a graphic, or some other kind of object:

```
if word 1 of the target is "graphic" then beep 3
```

Important! If the hcAddressing property of the stack containing the handler is set to true, the target function returns the names of controls preceded by the word `card` or `background`. In this case, you must check the second word of the returned value instead of the first.

How to determine whether a file exists

See Also:

How to create a file, How to delete a file, How to rename a file, Recipe for checking whether a volume or folder is write-protected, defaultFolder property, files function, folders function, there is a operator, URL keyword

You find out whether a file or folder exists using the there is a operator. The following statement determine whether there is a file called "My File" in a folder called "The Folder", which resides on a disk named, unimaginatively, "Disk":

```
if there is a file "/Disk/The Folder/My File"
then get URL "binfile:/Disk/The Folder/My File"
else
  beep
  answer "The file was not found."
end if
```

How to determine whether a file path is relative or absolute

See Also:

How to include a slash in a pathname, About filename specifications and file paths, Recipe for converting an absolute path to a relative path, Recipe for converting a relative path to an absolute path

You check whether a file path is a relative path (that is, a path that starts from the current folder) or an absolute path (that is, one that starts from the top level of the file system and doesn't depend on which folder is the current folder) by checking the first characters in the path.

On Mac OS, OS X, and Unix systems, an absolute path starts with a slash (/). If a path starts with any other character, it is a relative path.

On Windows systems, an absolute path starts with a drive letter (A, B, C, etc.) followed by a colon (:). If the second character is not a colon, the path is a relative path.

How to determine which line of a list field is selected

See Also:

How to select an entry in a list field, Why can't I leave all lines of a list field unselected?, hilitedLine property, listBehavior property, selectedText function

You find out which line (or lines) in a clickable list field is currently selected using the field's hilitedLine property. The following statement selects the line after the selected line:

```
put the hilitedLine of field "My List" into myVar  
set the hilitedLine of field "My List" to myVar + 1
```

If the field's multipleHilites property is true, the user can select multiple lines. In this case, the hilitedLine reports all the selected line numbers, separated by commas.

Tip: The selectedText function reports the actual text of the selected line or lines.

How to determine which pending messages have been scheduled


See Also:

How to cancel a pending message, How to schedule a future message, cancel command, pendingMessages function, send command

You use the pendingMessages function to obtain a list of events that have been scheduled with the send...in time form of the send command. You can also use the Pending Messages mode of the message box.

When you use the send command with a time delay, it returns a message ID number in the result, and adds a line to the pendingMessages containing the message ID number, scheduled time, message name, and target object. You can find out whether any of a particular message have been scheduled with a script line such as the following:

```
if "myMessage" is among the items of the pendingMessages
```

To see a list of pending messages, click the  (envelope) icon in the message box to show the Pending Messages list.

How to disable a tab in a tabbed button

See Also:

A tabbed button is implemented as a menu with each tab's name being one line in the button's content so you enable or disable tabs the same way you enable and disable menu items in a button menu.

You enable or disable a tab by using the enable menu or disable menu command in a handler or the message box. The following statement disables the second tab of a tabbed button:

```
disable menuItem 2 of button "Tab Choices"
```

Tip: Placing an open parenthesis (before the tab's name is equivalent to using the disable menu command. Removing a leading open parenthesis is equivalent to using the enable menu command. You can change leading parentheses by setting the tabbed button's text property, or in its property inspector.

How to disable tool tips

See Also:

How to get information about items on the screen, `toolTipDelay` property, `toolTip` property, Edit menu > Preferences

You prevent the display of tool tips by setting the `toolTipDelay` property to zero, using the following statement in a handler or the message box:

```
set the toolTipDelay to zero
```

The `toolTipDelay` controls how long the user must point to an object before its tool tip appears. If the `toolTipDelay` is zero, the tool tips are disabled.

To disable tool tips in the Revolution development environment, choose Edit menu>Preferences. In the "Documentation" pane, uncheck the "Display Tool Tips" box.

How to display a dialog box

See Also:

How to close a modal dialog box, How to close a window, How to create a custom dialog box, How to edit a palette or dialog box, Why doesn't the ask or answer dialog appear in my standalone?, Recipe for Hello World, Recipe for speaking an alert message, answer command, answer file command, answer folder command, answer printer command, ask command, ask file command, ask folder command, dialogData property, modal command, modeless command, style property

You display a simple dialog box by using the ask or answer commands in a handler or the message box. Use the ask command if you want to request text from the user. Use the answer command to display a message or to ask the user to make a choice.

Tip: You can also create a custom dialog box containing any objects you want to include. To create a custom dialog box, create a stack with the controls you want (including at least one button with a mouseUp handler that closes the stack). Then use the modal command to display the stack as a dialog box.

How to display a picture from a web server

See Also:

How to access the Internet from behind a firewall, How to import a picture file into an existing image objects, How to use a password for a web page, About containers, variables, and sources of value, About using URLs, uploading, and downloading, Image Types Reference, Why does an image disappear?, filename property

To display a picture file thatÆs on a web server, you create an image, then set the imageÆs filename property to the URL of the picture you want to display. Whenever the image is displayed, Revolution downloads the picture from the server and displays it in the image.

To instead download the image and make it a permanent part of the stack, put the content of the URL into the image, as in the following example:

```
put URL "http://www.example.net/picture.gif"
    into image "My Image"
```

Note: The first method ensures that the image reflects changes made to the picture on the server. However, it causes a delay whenever you go to a card that has the image on it, while Revolution downloads the picture file.

How to display a single record from an automatic database query

See Also:

How to display records from an automatic database query as a table, How to navigate among records from an automatic database query, How to save changes to records from an automatic database query, How to set up an automatic database query, About connecting to and using SQL databases, Tools menu > Database Query Builder, revDataFromQuery function

You use the Database pane in a field's Properties palette to display a result from an automatic database query in the field. Follow these steps:

1. Select the field and choose Object menu>Object Inspector, then choose "Database" from the menu at the top of the inspector.
2. From the Query menu, choose the automatic query you want to use.
3. From the Column menu, choose the name of the database field you want to display. That database field's contents from the current record is displayed in the field.

You can create fields to show all the database fields you want to display in the current record.

How to display a splash screen

See Also:

Most applications display a splash screen—a window that shows the application name, version, author, and perhaps other information—when the application starts up. The splash screen automatically closes when the application is ready to use.

There are several methods for displaying a splash screen. One of the simplest is to create a stack to hold the splash screen. In your application's startup handler, open the splash screen. (You can either hide it again after a specified period of time, or use a `mouseDown` handler in the splash screen stack to hide it when the user clicks the screen.)

Tip: If you set the splash screen stack as your application's main stack, the splash screen opens automatically when your application launches, with no need to explicitly show it.

How to display a TIFF file

See Also:

File menu > New Referenced Control > QuickTime-Supported File...

You display a TIFF file in a stack window by creating a player and setting the player's filename property to the TIFF file's name. (Since QuickTime supports the TIFF file format, you can display a TIFF file in a player, even though it's not a video or sound.)

You display a file in a player by choosing File menu>New Referenced Control>QuickTime-Supported File. In a handler, first use the create command to create the player, then set its filename property to the name and location of the file you want to display.

Tip: Make sure the player's showController property is set to false, since the QuickTime controller doesn't make sense when displaying a picture.

How to display a web page in a field

See Also:

How to access the Internet from behind a firewall, How to open a URL in a browser, How to retrieve headers from an HTTP request, How to use a password for a web page, Recipe for switching between styled text and HTML source, `htmlText` property, `http` keyword, `load` command, `URL` keyword

You display a web page in a field by setting the field's `htmlText` property to the web page's URL.

To display a web page, use a statement like the following in a handler or the message box:

```
set the htmlText of field "Show Web Page"
to URL "http://www.example.org/index.html"
```

You may have to wait a moment for the web page to be downloaded.

Note: The `htmlText` property does not implement many HTML features, such as tables and frames. Revolution supports a subset of HTML tags that includes font, size, style, and text color information, links, and image references.

How to display an icon or picture in a button

See Also:

How to import a picture file into an existing image object, Why do icons disappear from a standalone application?, armedIcon property, disabledIcon property, hilitedIcon property, icon property, showIcon property

You display a picture in a button by creating an image object to hold the picture, then setting the button's icon property to the image's short ID. The button's showIcon property must also be set to true.

The icon can be an image of any size, and the image can either be a referenced control displaying a file, or an image that's been imported into the stack or created with the paint tools. The image does not need to be in the same stack as the button and does not need to be visible in order to be displayed in the button.

You can either set this property in a handler or the message box, or use the Basic Properties pane in the button's property inspector to set a button's icon to an image from Revolution's built-in icon library.

How to display an object's property inspector

See Also:

How to edit an object's script, About custom properties and custom property sets, About properties and property profiles, Shortcut to display a contextual menu, Shortcut to display a contextual menu (Pointer tool), Tools menu > Application Browser, Object menu > Object Inspector, Object menu > Card Inspector, Object menu > Stack Inspector

You display an object's property inspector by selecting it with the Pointer tool or by using the contextual menu shortcut. To display the property inspector, do any of the following:

- ò Select the object with the Pointer tool and either choose Object menu>Object Inspector, or click the Object Inspector icon on the Toolbar.

- ò Double-click the object with the Pointer tool.

- ò Command-Control-Shift-click the object (Mac OS or OS X) or Control-Shift-right-click (Unix or Windows) with the Browse tool, then choose the Property Inspector from the contextual menu.

- ò Choose Tools menu>Application Browser, find the object, and double-click its name.

How to display another stack in the current window

See Also:

How to bring a window to the front, How to change the order of cards in a stack, How to go to another card, How to open a stack, Why does an object or window flicker?, go command

To open a stack in another stack's window, you use the go command while specifying the window you want to use. For example, if a stack named `ôHateö` is open, the following statement opens the stack `ôLoveö` in the same window:

```
go stack ôLoveö in window ôHateö
```

If the two stacks have the same height and width, there is no visible change in the window (apart from the fact that the contents and window title may change). You can use this command to make two different stacks appear to be the same stack to the user.

How to display objects on more than one card

See Also:

How to select a group, How to select a control, Object menu > Group Selected, Object menu > Place Group, backgroundBehavior property, group command, place command, remove command

You display the same object (or a set of objects) on more than one card by making them part of a group. Select the object or objects you want to group, then choose Object menu>Group Selected. To do this in a handler, use the group command.

Once you've grouped the objects, go to each card you want them to appear on, and choose the group from the "Place Group" submenu in the Object menu. (You may want to give your group a descriptive name first, if you're using more than one.) To place a group on a card from within a handler or the message box, use the place command.

Tip: To place a group automatically whenever you create a new card, set the group's backgroundBehavior property to true. If you are on a card that has the group when you create the new card, the group is placed on the new card.

How to display records from an automatic database query as a table

See Also:

How to display a single record from an automatic database query, How to set up an automatic database query, About connecting to and using SQL databases, Database Types Reference, Tools menu > Database Query Builder, revDataFromQuery function

You use the Database pane in a field's Properties palette to display the records from an automatic database query in the field. Follow these steps:

1. Select the field and choose Object menu>Object Inspector, then choose "Database" from the menu at the top of the inspector.
2. From the Query menu, choose the automatic query you want to use.
3. From the Column menu, choose "Show All".
4. Choose "Table" from the menu at the top of the property inspector, then click the box labeled "Table object" to display the records as a grid.

Note: If you change the automatic query in the Database Query Builder, the field's content is updated automatically.

How to display text on a card

See Also:

How to display the same text on multiple cards, How to prevent changing a field's text, Why can't I select, copy, or paste text?, Object menu > Object Inspector, Object menu > New Control > Label Field, lockText property, showBorder property

You display text on a card by putting it into a field object.

To create a field that displays only static text, set the field's lockText property to true and its traversalOn, opaque, and showBorder properties to false. You can either set these properties in a handler or the message box, or use the Basic Properties pane in the field's property inspector.

Tip: To create a field that is pre-set to display static text, choose Object menu>New Control>Label Field. Then use the Contents pane in the field's property inspector to edit the text.

How to display the contents of a field in a printed report

See Also:

How to print a report, How to create a report layout card, Tools menu > Report Builder, Object menu > New Control > Report Object, View menu > Go to Report Page

To display a field in a printed report, you create a report viewer object on the report layout card, then link it to the field you want to display. To create a report viewer, follow these steps:

1. Choose Object menu>New Control>Report Object.
2. In the new report viewer's property inspector, enter a field reference in the box labeled "Source", or click the Choose () button to choose a field from any open stack.
3. Click Apply to link the field's contents to the report viewer.

When you print the report, the content of the linked field is substituted for the report viewer for each of the cards to be printed.

How to display the contents of a text file

See Also:

How to display text on a card, How to export data to a file, How to import a text file into a field, How to put text into a field, About containers, variables, and sources of value, About using URLs, uploading, and downloading, Recipe for reading the contents of a file, File menu > Import As Control > Text File..., file keyword

You display a text file by putting the file's URL into a field. The URL can be local (a file URL) or a reference to a file on the Internet (an ftp or http URL).

The following example creates a field and imports a file into it:

```
create field "Text"  
put URL "file:/Disk/Folder/text.txt" into field "Text"
```

Tip: To easily create a new field that holds the text of a file, choose File menu>Import As Control>Text File.

How to display the paint tools

See Also:

How to copy an image to the clipboard, Shortcut to magnify an image, choose command

You use the contextual menu shortcut to display a palette of paint tools you can use to create and change images.

To display the paint tools, Command-Control-Shift-click (Mac OS or OS X) or Control-Shift-right-click (Unix or Windows) an image. In the contextual menu that appears, choose "Paint Tools" to show the paint tools palette. Choose any of the paint tools to work with the image.

When you're done, choose the Browse tool again.

How to display the print settings dialog box

See Also:

How to print a card, How to print in landscape mode, How to print the contents of a field, File menu > Page Setup..., answer printer command, print command, revPrintField command, revPrintText command, revShowPrintDialog command

You use the answer printer command to display the print settings dialog box (Page Setup on Mac OS systems). Using the answer printer command lets the user change print settings before your script issues the open printing or print commands.

If you are using the revPrintText or revPrintField command, use the revShowPrintDialog command instead of answer printer.

Cross-platform note: To display the Print dialog box on Mac OS systems, use the open printing with dialog form of the open printing command.

How to display the same text on multiple cards

See Also:

How to display text on a card, How to give a checkbox a different state on each card, `sharedText` property

If a field is part of a group, and that group is placed on more than one card, the field can either display the same text on all cards, or display different text on each card where the field appears.

You make a grouped field display the same text on each card by setting its `sharedText` property to `true`.

Note: A group can consist of a single control, so you can have a group placed on multiple cards that consists only of the field whose text you want to display.

How to distribute a stack to users on other platforms

See Also:

How to create a standalone application, How to preview how a stack will look on other platforms, Supported Platforms Reference, File menu > Build Distribution..., charSet property, platform function

To distribute a stack, you simply let other users copy or download it. In general, no changes to your stack are required. If your users have any edition of Revolution, they can use the stack as-is. Otherwise, you can build a standalone application for each target platform.

Tip: When you open a stack on a Mac OS or OS X system that was last saved on a Unix or Windows system (or vice versa), the text in the stack is translated automatically to the appropriate character set. The process can take a perceptible amount of time, so it's a good idea to save a stack destined for a particular platform on that platform before delivering it to users.

How to do a task when starting up the application

See Also:

How to respond to double-clicking a file for Mac OS or OS X, How to respond to double-clicking a file for Windows, How to respond when a card is visited, How to respond when a stack opens, startup message

To perform an action when the application starts up, you handle the startup message.

The startup message is sent when the application is first starting up, to the first stack opened. When you double-click a standalone application, the startup message is sent to the standaloneÆs main stack.

Tip: To perform an action when the development environment starts up, create and install a plugin stack whose openStack handler contains the actions you want to do. Then choose Development menu>Plugins>Plugin Settings and set the plugin to open when Revolution starts up.

How to download a binary file

See Also:

How to cancel a file transfer in progress, How to download a file from an FTP server, How to list the files in an FTP directory, How to upload a file to an FTP server, How to use a password for an FTP server, About using URLs, uploading, and downloading, Why was a downloaded file corrupted?, binfile keyword, file keyword, libURLDownloadToFile command, URL keyword

To download a binary file, you use the libURLDownloadToFile command, or else put the file's URL into a binfile URL. The following examples both download the same binary file:

```
libURLDownloadToFile "http://ftp.example.net/some.pdf",
```

```
    "/Disk/Folder/New File.pdf" -- is equivalent to...  
put URL "ftp://ftp.example.net/some.pdf"
```

```
into URL "binfile:/Disk/Folder/New File.pdf"
```

Important! You should always refer to a binary file with a binfile URL. If you use a text URL, Revolution automatically translates the end-of-line marker during the transfer. However, for binary files, changing the end-of-line marker will corrupt the file, so Revolution leaves the file data untouched when you use a binfile URL.

How to download a file from an FTP server

See Also:

How to cancel a file transfer in progress, How to download a binary file, How to list the files in an FTP directory, How to upload a file to an FTP server, How to use a password for an FTP server, About using URLs, uploading, and downloading, Why was a downloaded file corrupted?, `binfile` keyword, `file` keyword, `libURLDownloadToFile` command, `URL` keyword

To download a file from an FTP server, you use the `libURLDownloadToFile` command, or else put the file's URL into a container (which can be another URL):

```
put URL "ftp://ftp.example.com/newfile.dat"

into URL "binfile:newfile.dat"
```

Tip: Getting the content of an ftp URL causes the handler to pause until the download is complete, but the `libURLDownloadToFile` command does not pause, and the `libURLDownloadToFile` command does not require that the entire content of the file fit into memory, so it is preferable for downloading large files.

How to draw a shape

See Also:

How to create a control, Object menu > New Control > Rectangle Graphic, Object menu > New Control > Oval Graphic, Object menu > New Control > Curve Graphic, Object menu > New Control > Round Rect Graphic, Object menu > New Control > Polygon Graphic, Object menu > New Control > Line Graphic, Object menu > New Control > Regular Polygon Graphic, graphic keyword, points property, style property

You draw a shape using the Graphic tool. The style of the graphic you draw determines its shape (curve, oval, rectangle, round rectangle, or polygon), so you can change a graphic's shape by setting its style.

To draw a graphic by hand, click one of the graphic icons on the Tools palette and drag out the shape you want.

In a handler, you can set the style of the `templateGraphic` to the graphic type you want, then draw the graphic with the Graphic tool. (The graphic icons on the Tools palette work this way: each one sets the style of the `templateGraphic`, then chooses the Graphic tool so you can create the graphic.)

How to duplicate a custom property set

See Also:

How to create a custom property set, How to delete a custom property set, How to rename a custom property set, About custom properties and custom property sets, customProperties property, customPropertySet property, customPropertySets property

You duplicate a custom property set using the customProperties property. You may want to do this if, for example, you have created a set of prompts in one language and want to translate it into a second language that uses a second custom property set.

The following example makes a copy of a custom property set called `EnglishPrompts`:

```
set the customProperties["ItalianPrompts"] of button 2  
to the customProperties["EnglishPrompts"] of button 2  
-- creates the new custom property set "ItalianPrompts"
```

The new `ItalianPrompts` custom property set contains the same custom properties and values as the original set `EnglishPrompts`.

How to duplicate an object

See Also:

How to copy a card, How to move cards from one stack to another, Shortcut to duplicate a control, Edit menu > Copy, Edit menu > Duplicate, Edit menu > Replicate..., clone command, copy command

You duplicate an object by selecting it and choosing Edit menu>Duplicate, or by using the clone command in a handler or the message box. (The clone command can be used to duplicate cards and stacks, as well as controls.)

To make multiple duplicates of a control, choose Edit menu>Replicate instead.

Tip: Instead of duplicating the object, you can instead select it, copy it, and paste it. After copying, go to the destination you want to place the object in, then either choose Edit menu>Paste, or use the paste command. The object is pasted onto the current card.

How to edit a palette or dialog box

See Also:

How to close a modal dialog box, How to create a custom dialog box, How to create a palette, How to display a dialog box, How to edit objects in a stack window, Shortcut to display a contextual menu, modal command, modeless command, palette command, style property, topLevel command

Revolution's editing tools (the Pointer tool, the "New Control" submenu of the Objects menu, and so on) work only in editable windows, not in palettes or modal or modeless dialog boxes.

You use the topLevel command to change the stack temporarily to an editable window so that the editing tools will work:

```
topLevel "My Stack"
```

Tip: If you don't know the stack's name, enter the following in the message box, then move the mouse pointer over the stack and press the Return key:

```
topLevel the mouseStack
```

How to edit an objectÆs script

See Also:

How to display an objectÆs property inspector, How to pretty print a script, About the structure of a script, Recipe for listing all the handlers in a script, Object menu > Object Script, Object menu > Card Script, Object menu > Stack Script, Shortcut to display a contextual menu, Shortcut to display a contextual menu (Pointer tool), Shortcut to edit a controlÆs script, Shortcut to edit the card script, Shortcut to edit the selected controlÆs script, Shortcut to edit the stack script, edit script command

You edit an objectÆs script in the script editor window. You can open the script editor by selecting the object and choosing Object menu>Object Script.

To open the script editor directly in a handler or the message box, use the following statement:

```
edit script of object
```

To change an objectÆs script in a handler or the message box without opening the script editor and manually entering the change, use the set command.

Tip: You can also use several shortcuts to open an objectÆs script. See the Shortcuts section of the documentation for more.

How to edit objects in a stack window

See Also:

How to edit a palette or dialog box, Tools menu > Pointer Tool, Tools menu > Tools Palette, Tools menu > Application Browser

Controls, such as fields and buttons, are displayed in a stack window. You move, resize, and change controls using the Pointer tool.

To use the Pointer tool, click the arrow icon at the top right of the Tools palette, or choose Tools menu>Pointer Tool. With the Pointer tool, click the object you want to work with to select it. You can drag the object to a new location, or drag the square handles at the corners to resize it. Choose Object menu>Object Inspector to change the object's settings.

Tip: You can use the Pointer tool only in a normal editable window, not in a dialog box or palette. To use the Pointer tool in one of these window types, change it temporarily to an editable window.

How to enable or disable a control

See Also:

How to enable or disable a menu item, Why can't I select, copy, or paste text?

You enable or disable a control by setting the control's enabled or disabled properties. (These properties are inverses of each other. Setting one to true automatically sets the other to false, and vice versa.)

When a control is disabled, its appearance is dimmed and it no longer responds to user actions such as clicking. Disabled fields cannot be changed, disabled buttons do not highlight when clicked, and disabled scroll bars cannot be scrolled.

You can either set these properties in a handler or the message box, or use the Basic Properties pane in the control's property inspector to disable or re-enable the control.

How to enable or disable a menu item

See Also:

How to enable or disable a control, How to show or hide the menu bar, About menus and the menu bar, disable menu command, disabled property, enable menu command, enabled property

You enable or disable a single menu item in any menu by using the enable menu or disable menu command in a handler or the message box. The following statement disables the second menu item in the Style menu:

```
disable menuItem 2 of menu "Styles"
```

Note: If the menu is not in the menu bar, use ..of button... instead of ...of menu....

Tip: Placing an open parenthesis (before the menu itemÆs text is equivalent to using the disable menu command. Removing a leading open parenthesis is equivalent to using the enable menu command. You can change leading parentheses when setting the menuÆs text property, or in a button menuÆs property inspector.

How to enter or display Unicode text in a field

See Also:

How to convert between Unicode and ASCII text, How to find out whether text in a field is Unicode, How to import a Unicode text file, `fontLanguage` function, `textFont` property, `unicodeText` property

You display double-byte text in its correct language by setting its `textFont` property to a Unicode font. You can either put the text into the field and set the `textFont` in a handler or the message box, or manually enter the text after using the operating system's built-in text entry tools to choose a language.

For example, to display double-byte Japanese characters that are on line 12 of a field, use a statement like the following:

```
set the textFont of line 12 of field 1 to "Osaka,Japanese"
```

When you manually enter text in a language that does not use the Roman alphabet, using the operating system's tools, Revolution automatically sets the `textFont` of the text you enter to the appropriate font for the language you have chosen.

How to enter the debugger

See Also:

How to prevent a handler from executing, How to prevent the debugger from appearing, How to temporarily remove a portion of a script, Debug menu > Set Breakpoint (Script Editor), Debug menu > Clear All Breakpoints (Script Editor)

You enter the debugger either by clicking `Debug` in the error window or by setting a breakpoint at the point in a handler where you want the debugger to appear.

To set a breakpoint, open the script and select the statement where you want the debugger to appear, then choose `Debug menu>Set Breakpoint`. A red indicator appears in the left margin of the script editor next to the line. When the statement is executed, execution is suspended and the debugger appears.

To remove the breakpoint, click in the line with the breakpoint, then choose `Debug menu>Set Breakpoint` again. To remove all the breakpoints in a script, choose `Debug menu>Clear All Breakpoints`.

How to establish a default value for a parameter

See Also:

About commands and functions, About containers, variables, and sources of value, Recipe for an approximate-equality function, Recipe for building a repeated string, Recipe for converting between characters and ASCII values, params function

In some situations, you may want to use a custom command or custom function without specifying all its parameters, and use a default value for those that you haven't specified.

If you leave a parameter out when you use a custom command or custom function, the handler that implements the command or function receives an empty parameter.

To substitute the default value, you check whether the parameter is empty in the custom message handler or function handler. If it is, then no value has been passed, and you can simply put the desired default into the parameter.

How to estimate how much memory your application will need

See Also:

Memory and Limits Reference, About Revolution system requirements, Why am I running out of memory?, Why is Revolution taking more memory than its allocation?, hasMemory function, heapSpace function

The amount of memory required by your standalone application depends on many factors, including the platform, the complexity of your code, which Revolution custom libraries you include, and how many windows you have open at once.

As a general rule of thumb, add up the size of the stacks to be loaded into memory (and the picture and movie files to be displayed in referenced controls) at any one time and add 4M to obtain a rough approximation:

	total size of loaded stacks
+	total size of external files displayed
+	4 megabytes
=	minimum memory required (very approximate)

How to evaluate an expression

See Also:

How to execute a command, Tools menu > Message Box, () operator, call command, value function

You evaluate an expression by entering the expression into the message box, or by using it in a handler.

To evaluate an expression immediately in the message box, enter the expression and press Enter or Return to evaluate it. The resulting value is placed in the message box. For example, entering this line into the message box produces the number 4:

$$1 + 1 * 2 + 2 / 2$$

To force an expression in a handler to be evaluated, use the value function, or enclose the expression in parentheses.

How to execute a command

See Also:

How to evaluate an expression, Tools menu > Message Box

You execute a command by entering the command name, along with any parameters it requires, into a handler or into the message box.

To execute a command immediately in the message box, enter the command and press Enter or Return to execute it. Any result of the command is placed in the message box. For example, entering this line into the message box causes the computer to beep twice:

```
beep 2
```

How to execute a handler one line at a time

See Also:

How to enter the debugger, How to monitor the value of variables while debugging, Debug menu > Step Into (Script Editor), Debug menu > Step Over (Script Editor), Debug menu > Trace (Script Editor)

You use the debugger to execute a handler one line at a time. During execution, you can use the variable watcher to monitor variable values as each line is executed, as well as watching whatÆs happening on the screen.

Once an executing handler has entered the debugger, click "Step Into" at the bottom of the debugger window (or choose Debug menu>Step Into) to execute the next statement. If the next statement calls another handler, the debugger enters that handler, and you can continue executing it line by line until control returns to the original handler.

Tip: To skip over called handlers, click "Step Over" instead of "Step Into".

How to execute a script statement dynamically

See Also:

How to execute a command, How to execute Transcript code in a file, How to find out whether a command succeeded, do command

Sometimes it's necessary to execute a piece of Transcript code that's not in a script. This code might be in a file, a variable, a custom property, or a field.

You use the do command to execute a statement or statements that aren't in a script. The following simple example asks the user for a single line of code and executes that line:

```
ask "Please enter a valid statement:"  
do it
```

Tip: Revolution uses this method to execute lines of code that you enter in the message box.

How to execute a SQL statement on a database

See Also:

How to make changes to a SQL database, How to save changes to a SQL database, About connecting to and using SQL databases, revCommitDatabase command, revExecuteSQL command, revRollBackDatabase command

You use the revExecuteSQL command to execute a SQL statement (other than a SELECT statement).

When you open a database connection, the revOpenDatabase function returns an ID number, which you use to refer to that database. The following example creates a new table in the database whose database ID number is 2:

```
revExecuteSQL 2,"CREATE TABLE NewOrders"
```

Note: To execute a SELECT statement, use the revQueryDatabase function instead.

How to execute Transcript code in a file

See Also:

How to create a code library, How to execute a script statement dynamically, do command, file keyword, URL keyword

You execute Transcript code in a file using the do command.

The following example executes a statement or statements that are stored in a file called `Code.txt`:

```
do URL "file:/Disk/Folder/Code.txt"
```

You can execute a single statement, a list of statements, or a set of statements that include control structures (such as repeat and if).

How to export data to a file

See Also:

How to append data to the end of a file, How to create a file, How to import data from a file, How to store preferences or data for a standalone application, About filename specifications and file paths, About using URLs, uploading, and downloading, binFile keyword, file keyword, put command, resFile keyword, URL keyword, write to file command

You export data to a file by putting it into a file URL.

For example, to place the contents of a field into a file called `test.txt` in the current folder, use the following statement in a handler or the message box:

```
put field 3 into URL "file:test.txt"
```

Tip: To export binary (non-text) data, use the `binfile` URL scheme instead of the `file` URL scheme.

Tip: To work with a file too large to fit into available memory, use the `write to file` command instead of using the `put...into URL` form.

How to export text from a field in RTF format

See Also:

How to import and export styled text, How to import an RTF file, file keyword, htmlText property, RTFText property, URL keyword

You use the RTFText property to export styled text in RTF format.

The following statement exports the text in a field to an RTF file:

```
put the RTFText of field "Text" into URL "file:test.rtf"
```

How to fetch data from the Internet

See Also:

How to access the Internet from behind a firewall, How to display a picture from a web server, How to display a web page in a field, How to download a file from an FTP server, How to open a URL in a browser, How to use a password for a web page, How to use a stack thatÆs on a web server, About using URLs, uploading, and downloading, Why canÆt I open a downloaded stack?, ftp keyword, http keyword, libURLDownloadToFile command, load command, open socket command, URL keyword

You get data, such as a file, from a web server or FTP server by using the fileÆs URL in an expression.

For example, to place the contents of a file from a web server in a field, use a statement like the following:

```
put URL "http://www.example.net/myfile.txt"

into field "File Contents"
```

Revolution downloads the content of the URL you specify, and places that content in the field.

You can place the contents of a URL in a variable, field, or any other container.

How to find out the actual color of an object

See Also:

How to change an object's color, How to change the color of text, How to restore the default colors of an object, About colors and color references, Why do fonts and colors change when I create a standalone application?, Why don't buttons respect my color settings?, effective keyword

You find the color of an object using the effective keyword along with a color property.

If one of an object's colors is not specified—that is, if the corresponding color property is empty—then it inherits the color of its owner. For example, if a button's `foregroundColor` is set to empty, it uses the `foregroundColor` of the card it belongs to. If the card's `foregroundColor` is empty, it uses the `foregroundColor` of its stack. To examine the button's owners and find out what `foregroundColor` is actually displayed, use a statement such as the following:

```
get the effective foregroundColor of button "My Button"
```

Tip: You can use the effective keyword with any inherited property.

How to find out the highlight state of a checkbox or button

See Also:

How to change the highlight state of a checkbox or button, How to give a checkbox a different state on each card, autoHilite property, hilite command, hilite property, radioBehavior property, unhilite command

The state of a control such as a radio button or checkbox is specified by the control's hilite property. If the hilite is true, the button is checked or turned on. If the hilite is false, the button is unchecked or turned off.

The following example sets a variable depending on a checkbox's hilite property:

```
if the hilite of button "Show Results" is true  
then put true into doShowResults
```

How to find out the location of the current stack's file

See Also:

About filename specifications and file paths, Why can't I find a stack I just saved?, defaultFolder property, filename of stack property, stackFiles property

You use the filename of stack property to determine the name and location of a stack file. If the stack you want to check is a main stack, its filename of stack property reports the file path of the stack file:

put the filename of stack "My Main" into thePath

If the stack is a substack, use the effective keyword:

put the effective filename of stack "Sub" into thePath

Tip: To place the current stack's file path in a variable, enter the following in a handler or the message box:

set the itemDelimiter to "/"

get item 1 to -2 of the effective filename of this stack

How to find out the number of elements in an array

See Also:

You use the keys function to find out the number of elements in an array variable:

```
put the number of lines in the keys of myArray
```

```
into numberOfElements
```

The keys function returns a list of elements, one per line, so the number of elements in the array is the same as the number of lines in the keys.

How to find out the object type of an object

See Also:

How to determine what kind of object triggered a message, How to list the objects on a card, How to refer to a newly-created control, About object types and object references, control keyword, HCAddressing property, name property

You find out the object type of an object (whether the object is a field, button, stack, etc.) by checking the object's name property. The first word of an object's long or abbreviated name is its object type.

The following example checks whether the object under the mouse pointer is a player:

```
if word 1 of the abbr name of the mouseControl is "player"  
then beep  
else startMovingControl
```

Note: If the HCAddressing property of the stack is set to true, the object type is the second word of the long or abbreviated name, not the first.

How to find out what text the user clicked

See Also:

How to find out where the user clicked, clickChunk function, clickText function, link keyword

You find out what text the user clicked in a field using the clickText function.

When the user clicks text in a locked field, Revolution sets the clickText function to the word the user clicked. (If the text's textStyle is `link`, the clickText is the whole run of grouped text, not only the clicked word.) You can use this function in a mouseDown handler, as in the following statement:

```
go card the clickText -- the card whose name was clicked
```

Tip: To obtain a chunk expression specifying what text the user clicked instead of the text itself, use the clickChunk function instead.

How to find out where the insertion point is

See Also:

How to determine which line of a list field is selected, How to find out what text the user clicked, How to find out where the mouse pointer is, How to place the insertion point in a field, selectedChunk function, selection keyword

You use the selectedChunk function to find out the location of the insertion point.

When there is an insertion point but no text is selected, the selectedChunk function returns an expression of the form char nextChar to prevChar of field fieldNumber. For example, if the insertion point is in field 3, between character 10 and character 11, the selectedChunk function returns char 11 to 10 of field 3.

How to find out where the mouse pointer is

See Also:

How to find out where the user clicked, How to simulate a mouse click, mouseLoc function, screenMouseLoc property

In a handler, you find out where the mouse pointer is by using the mouseLoc function.

Tip: The mouseLoc function returns the position of the mouse pointer relative to the current stack. To find the position of the mouse pointer relative to the screen, use the screenMouseLoc property instead.

To find out which object the mouse is over, you use the mouseControl function.

How to find out where the user clicked

See Also:

How to find out what text the user clicked, How to find out where the mouse pointer is, How to find out which object the user clicked, How to simulate a mouse click, clickLoc function, clickText function, mouseLoc function

In a handler, you use the clickLoc function to find out where the user last clicked. The following statement puts the location of the last click into a field:

```
put the clickLoc into field "Display Location"
```

Tip: To find out the location of the last click in global coordinates, use the globalLoc function, as in the following statement:

```
put globalLoc(the clickLoc) into myScreenLocation
```

How to find out whether a command or function is defined

See Also:

How to create a synonym for a command or function, About commands and functions, try control structure

You may need to determine whether a handler exists for a custom command or function. For example, you may not know ahead of time which version of a library will be used with a stack, and whether the current version defines a particular command.

To find out whether a command or function exists, you use it in a handler. If it doesn't exist, using it causes an execution error. To prevent the user from seeing the error, enclose the command or function call in a try control structure, as in the following example:

```
try
  doSomeCommand
catch theError
  -- do whatever is needed if doSomeCommand doesn't exist
end try
```

How to find out whether a command succeeded

See Also:

How to return an error message from a handler, About commands and functions, result function, try control structure

Some actions, such as creating a file, might succeed or fail depending on factors your application cannot control (such as whether there's enough disk space). When using such commands, it's a good idea to double-check whether they were executed successfully.

Many built-in commands place an error message in the result function if they don't successfully execute. You can check the result after the command in order to see whether the command succeeded:

```
open file "Hello.jpg" for write
if the result is not empty then -- failed
  answer "Couldn't open file."
  exit to top
end if
```

How to find out whether a container is an array

See Also:

About containers, variables, and sources of value, How to access a value in an array variable, How to create an array variable, keys function

You find out whether a container is an array or a plain value by checking the keys property. If the keys of a container is empty, it is not an array.

Use a statement like the following to check whether a variable is an array or not:

```
if the keys of myVariable is empty then doNotAnArray  
else doAnArray
```

How to find out whether a custom property exists

See Also:

How to create a custom property, How to delete all an object's custom properties, How to list an object's custom properties, About custom properties and custom property sets, Why doesn't a custom property appear in the property inspector?, customKeys property, customProperties property

You use the customKeys property of an object to find out whether the object has a particular custom property. You can also use the Custom Properties pane in the object's property inspector to see whether it has a custom property.

The customKeys property lists all of an object's custom properties, one per line. This example checks whether a certain button has a custom property called `myProp`:

```
if "myProp" is among the lines of the customKeys  
  of button "OK" then beep -- custom property exists!
```

Important! If the object has more than one custom property set, you must set the customPropertySet to the correct set before checking the customKeys.

How to find out whether a global variable exists

See Also:

How to search global variables, About containers, variables, and sources of value, Development menu > Variable Watcher, global command, globalNames function

You find out whether a global variable has been declared using the Global Variables pane of the message box, or using the variable watcher, or with the globalNames function.

To display the variable watcher, choose Development menu>Variable Watcher. When the debugger is not running, the variable watcher window displays a list of all global variables that have been defined with the global command.

To check whether a global variable exists in a script, use the globalNames function:

```
if "myGlobal" is among the items of the globalNames then...
```

How to find out whether a window is open

See Also:

How to bring a window to the front, How to list all open windows, Why does Revolution ask to purge a stack?, Recipe for finding out whether a window is open by title, openStacks function

The openStacks function returns a list of open stack windows. To find out whether a window is open, you check the openStacks to see whether the stack's name is present. For example, to find out whether a window called "My Window" is open, use a statement like the following:

```
if "My Window" is among the lines of the openStacks...
```

The openStacks function returns the name of stack windows, not their label (which is what is shown in the window's title bar). To find out whether there is an open window with a certain title, you need to check the label property of each stack returned by the openStacks.

Tip: In the Revolution development environment, open windows are also listed in the Window menu. Choose a window to bring it to the front.

How to find out whether an image has anything in it

See Also:

An image is a container, whose contents is the picture in the image. If an image object exists, but has no painted pixels, the content of the image is empty:

```
if image "My Image" is empty  
then answer error "You must paint something in the image."
```

(This test does not apply to referenced images, whose picture content is stored in another file. It applies only if the image's filename property is empty, indicating it is not a referenced image.)

Tip: You can also use the image's text property to get its contents.

How to find out whether an image is a referenced image

See Also:

A referenced image is an image whose content is located in a separate file that is specified by the image's filename property. A non-referenced image's picture content is stored in the image object itself, and can be painted using the Revolution paint tools.

To find out whether an image is a referenced image, check its filename property. The filename of a non-referenced image is empty, since the content is not stored in a separate file:

if the filename of last image is empty then showPaintTools

How to find out whether OS X is running

See Also:

Since many operating-system details, such as where absolute file paths begin, differ between OS X and Mac OS, you may need to check which one is running. The platform function returns `Mac OS` for both Mac OS and OS X. To differentiate between the two, check the `systemVersion` function as well:

```
set the itemDelimiter to "."
if the platform is "Mac OS"

    and item 1 of the systemVersion >= 10
    then answer "OS X is running!"
```

Because the system version numbers of OS X started with `10.0` and the latest system version of Mac OS is `9.x`, checking the first part of the `systemVersion` lets you find out which of the two is running.

How to find out whether QuickTime is available

See Also:

Why don't movies play?, Why don't movies work on a Windows CD-ROM?, Why don't visual effects work?, dontUseQT property, dontUseQTEffects property, play command, QTVersion property

The QuickTime software is installed by default on Mac OS and OS X systems, and is always used on those systems to play video and sound. On Windows systems, QuickTime is used if it is installed.

To find out whether QuickTime is installed, you use the QTVersion function. If the function returns 0.0, QuickTime is not installed:

```
if the QTVersion is "0.0" then answer "No QT installed!"
```

How to find out whether text in a field is Unicode

See Also:

How to convert between Unicode and ASCII text, How to import a Unicode text file, charToNum function, effective keyword, fontLanguage function, numToChar function, textFont property, unicodeText property, uniDecode function, uniEncode function, useUnicode property

You find out whether text in a field is Unicode text by examining its textFont property. The textFont of Unicode text consists of the font name, a comma, and either `ôUnicodeö` or the language the text is in. The following example statement checks whether line 3 of a field is Unicode:

```
if the effective textFont of line 3 of field 1  
contains comma then answer "ItÆs Unicode!"
```

Note: Characters in chunk expressions are assumed to be single-byte characters. To check a Unicode characterÆs textFont using a chunk expression, treat it as two single-byte characters. For example, to check the fifth character in a field consisting of double-byte characters, use the expression the effective textFont of char 9 to 10 of field 1.

How to find out which card is the current card

See Also:

You address the card the user is on by using the descriptor this card:

set the backgroundColor of this card to "blue"

To get a reference to the current card, get its name, ID, or number property:

put the long ID of this card into myStoredName
answer "This is card" && the short name of this card
put the number of this card into field "Record Number"

To address the current card in a stack thatÆs not the current stack, add the stackÆs name:

get the number of this card of stack "Jewels"

How to find out which databases are open

See Also:

How to close the connection to a database, How to find out which record sets are open in a database, How to save changes to a database, About connecting to and using SQL databases, revCloseDatabase function, revOpenDatabase function, revOpenDatabases function

You use the revOpenDatabases function to find out which databases are open.

When you open a database connection using the revOpenDatabase function, the function returns a database ID number. Since more than one database connection can be open at a time, you use this ID number to specify which database you mean. The revOpenDatabases function returns the ID numbers of all open database connections.

The following example closes all open databases:

```
repeat for each item thisDB in revOpenDatabases()  
  revCloseDatabase thisDB  
end repeat
```

How to find out which object the user clicked

See Also:

How to find out what text the user clicked, How to find out where the mouse pointer is, How to simulate a mouse click, clickLoc function, mouseDown message, mouseLoc function, mouseUp message, target function

In a handler, you use the target function to find out where the message being handled was first sent. When used in a mouseDown or mouseUp handler, the target function returns the object that was clicked.

For example, if you click a card button, a mouseDown message is sent to the button, then (if the button has no mouseDown handler) goes on to the card. If the card has the following mouseDown handler, then clicking any object on the card (or the card itself) displays a dialog box with the object's name:

```
on mouseDown
  answer the name of the target
end mouseDown
```

How to find out which record sets are open in a database

See Also:

How to create a record set in a SQL database, How to find out which databases are open, How to move through the records in a record set (database cursor), How to use a SQL query to select records in a database, About connecting to and using SQL databases, revDatabaseCursors function

You use the revDatabaseCursors function to find out which record sets (database cursors) are open in a particular database.

When you create a record set using the revQueryDatabase or revQueryDatabaseBLOB function, the function returns a record set ID number. The revDatabaseCursors function returns the ID numbers of all open record sets for a database.

The following example closes all open record sets in the database whose ID number is 4:

```
repeat for each item thisSet in revDatabaseCursors(4)
  revCloseCursor thisSet
end repeat
```


How to find the Preferences folder

See Also:

How to change the current folder used for file operations, About file specifications and file paths, defaultFolder property, specialFolderPath function

You use the specialFolderPath function to obtain the location of the Preferences folder. The following statement puts the path to the Preferences folder into a variable:

```
put specialFolderPath("Preferences") into myPrefsPath
```

On Mac OS systems, the statement above returns the path to the Preferences folder inside the System Folder.

On OS X systems, the statement returns the path to the user's Preferences folder (inside the Library folder in the user's directory).

Tip: On Windows systems, use the expression specialFolder(26) to get the path to the Application Data folder, which can be used to store application preferences.

How to format numbers in a table field

See Also:

How to allow editing of individual cells in a table field, How to create a spreadsheet-style table, internet keyword, long keyword, short keyword, system keyword

To control the format of numbers in a table field, you specify a format, along with the cells you want to apply the format to, in the field's property inspector. To choose a format, follow these steps:

1. In the "Table" pane, check the "Cell formatting" box.
2. Choose a column number from both menus labeled "Format Column".
3. From the "Using" menu, choose the format you want to use. In the "With" box, enter a prefix or suffix, a number of decimal places, a percentage value, or "short", "long", "internet", or "system" for a date.
4. Press the Tab key, then click "Add" to apply the format.

Note: You can apply only one format to any particular cell.

How to get a striped background in OS X

See Also:

How to make a throbbing button, modal command, modeless command, palette command, style property, topLevel command

Under Aqua user-interface guidelines, dialog boxes and palettes have a pinstriped background pattern. In Revolution, if a stack is opened as a modal or modeless dialog box or a palette, the window background is automatically shown with this pattern by default. The `backgroundColor` and `backgroundPattern` properties of the stack (and any stacks above it in the object hierarchy) must be empty; otherwise the background pattern or color overrides the striped background provided by the operating system.

Editable windows are shown with a white background, which is the Aqua standard for document windows.

Important! To see the pinstriped background pattern, the application must be running on an OS X system, and the `lookAndFeel` property must be set to `Appearance Manager`.

How to get command-line parameters

See Also:

About containers, variables, and sources of value, \$ keyword

If you start up the application by typing its name on the command line, you can include parameters with it. For example, on a Unix system, you might start up an application by typing the following on the command line:

```
MyApp -h -t
```

To get command-line information from within the application, use the special variables \$0, \$1, \$2, and so on. If you started up the application using the example above, then \$0 holds the string `MyApp`, \$1 holds `-h`, and \$2 holds `-t`:

```
on startup
  if $1 is "-h" then show field "Help"
  if $2 is "-t" then answer the time
end startup
```

How to get information about items on the screen

See Also:

How to investigate code in the development environmentÆs windows, Edit menu > Preferences, toolTip property, toolTipDelay property

Almost every button, icon, and text field in the Revolution development environment includes a brief explanation, presented as a tool tip. You view these tool tips by holding the mouse pointer briefly over the item you want information about. The tool tip pops up in a small window next to the item.

Tip: You can control the display of tool tips using the "Documentation" pane in the Preferences window. Choose Edit menu>Preferences, then click "Documentation". Using the checkbox labeled "Display tool tips", you can turn tool tips on and off, and control how long you must wait before a tool tip is displayed.

How to get the contents of a database field

See Also:

How to get the contents of records in a SQL database, How to move through the records in a record set (database cursor), How to use a SQL query to select records in a database, About connecting to and using SQL databases, revDatabaseColumnCount function, revDatabaseColumnNamed function, revDatabaseColumnNames function, revDatabaseColumnNumbered function, revDatabaseColumnType function

Once you have created a record set (database cursor) in a database, you use the revDatabaseColumnNamed and revDatabaseColumnNumbered functions to get the contents of a specific field.

First, you use the revMoveToFirstRecord, revMoveToLastRecord, revMoveToNextRecord, and revMoveToPreviousRecord commands to go to the record you want. Next, you use the revDatabaseColumnNamed function (with the name of the database field) or the revDatabaseColumnNumbered function (with the field's number) to get the field's contents. The following example puts the contents of a database field named "CUSTNAME" into a variable named currCustName:

```
get revDatabaseColumnNamed(9,"CUSTNAME",currCustName)
```

How to get the contents of records in a SQL database

See Also:

How to display records from an automatic database query as a table, How to get the contents of a database field, How to use a SQL query to select records in a database, About connecting to and using SQL databases, revDataFromQuery function

You get information from a database using the revDataFromQuery function. This function takes a SQL query, finds the records that match the query, and returns the data from the records, without creating a record set (database cursor).

First, use the revOpenDatabase function to open a connection to the database you want to use. Then use the revDataFromQuery function to obtain the records you want. You can put the returned value in a variable or field, save it to a file, parse it to find specific database field values, and so on.

Tip: You can use the Database Linked pane in a field's property inspector to show record data from a database in the field, without scripting.

How to get the location of the user's home directory

See Also:

How to change the current folder used for file operations, How to determine whether a file exists, How to find out the location of the current stack's file, How to use environment variables, About filename specifications and file paths, Why can't Revolution find a file I specified?, \$ keyword, answer folder command, defaultFolder property, revMacFromUnixPath function, revUnixFromMacPath function

On OS X and Unix systems, each user has a home directory, which holds the files and folders that belong to that user. The environment variable \$HOME holds the path to the current user's home directory:

```
put $HOME into myUsersPath
```

Tip: To specify the home directory in a file path, use the ~ character. For example, the file path `~/list.txt` specifies a file called `list.txt` in the user's home directory. The file path `~/puffball/prefs` specifies a file called `prefs` in the home directory of user `puffball`.

How to give a border to a radio button cluster

See Also:

How to create a radio button cluster, label property, showBorder property, showName property

In a complex layout, it may be difficult to distinguish a set of related controls from other controls that are nearby. Many user-interface standards advise you to set apart a group of controls visually by surrounding it with a border and giving it a visible name.

You show a border around a radio-button group by setting the group's showBorder and showName properties to true. You can show a similar border around any set of controls by grouping the controls and setting the group's showBorder and showName.

How to give a checkbox a different state on each card

See Also:

How to change the highlight state of a checkbox or button, How to detect the highlight state of a control, How to display the same text on multiple cards, hilite property, sharedHilite property

If a checkbox button is part of a group, and that group is placed on more than one card, the checkbox can either display the same hilite state—checked or unchecked—on all cards, or display a different state on each card where the checkbox appears. You give a checkbox button the ability to show a different state on each card by setting its sharedHilite property to false.

Note: If you set the checkbox's hilite property on one card, it does not affect the setting on other cards where the checkbox's group appears. If you query the checkbox's hilite property in a handler, it reports the hilite on the current card. To find out the setting on another card, specify the card, as in the following statement:

get the hilite of button "Setting" of card 44

How to go to another card

See Also:

How to display another stack in the current window, How to open a stack, Tools menu > Application Browser, View menu > Go First, View menu > Go Prev, View menu > Go Next, View menu > Go Last, Shortcut to go to the next card, Shortcut to go to the previous card, go command

The most commonly-used ways to go to another card are the go command and the first four menu items in the View menu.

You can use the go command in a handler or the message box to go to any card in any stack.

You can use the Application Browser to go to any card in any stack thatÆs loaded into memory. Choose Tools menu>Application Browser, find the stack you want to go to, and click the disclosure arrow next to it to reveal a list of cards. Control-click (Mac OS or OS X) or right-click (Unix or Windows) the cardÆs name and choose ôGoö from the contextual menu. The stack window comes to the front with the card displayed in it.

How to hide other applicationsÆ windows

See Also:

How to keep a window on top of other windows, How to show or hide a window, Edit menu > Preferences, View menu > Backdrop, backdrop property, hide command, hideConsoleWindows property, hidePalettes property, mouseDownInBackdrop message, mouseUpInBackdrop message

You use the backdrop property to hide the windows of other applications. When the backdrop property is set to a color or pattern, that color or pattern is drawn behind your application so that other applicationsÆ windows are not visible.

You can set the backdrop property in a handler or the message box, or by choosing View menu>Backdrop.

To change the color that appears when you choose the menu item, choose Edit menu>Preferences and set a new color in the ôSizes and Appearanceö pane.

How to hide the Toolbar

See Also:

Shortcut to hide Revolution palettes, Tip: Shrinking the Toolbar, View menu > Toolbar Text, View menu > Toolbar Icons

You hide the Toolbar at the top of the screen by hiding its icons and text.

Choose View menu>Toolbar Text and View menu>Toolbar Icons to turn off both menu items. When both the icons and text are turned off, the Toolbar is not visible.

To show the Toolbar again, choose one or both menu items to display the icons, the text labels, or both.

How to import a HyperCard stack

See Also:

Why does importing a HyperCard stack cause an error?, File menu > Open Stack..., HCImportStat property, HCStack property

You import a HyperCard stack to Revolution by opening the stack with the `Open Stack` menu item in the File menu, or by using the `go` command in a handler or the message box. Revolution automatically converts the stack to its own stack format and checks the scripts for compile errors.

On Unix or Windows systems, you can import HyperCard stacks that have been converted to BinHex or MacBinary format and downloaded to your system.

Tip: Before importing a HyperCard stack, use HyperCard's `Compact Stack` menu item to remove extraneous information from it.

How to import a picture file into an existing image object

See Also:

How to display a picture from a web server, How to display a TIFF file, How to display an icon or picture in a button, How to take a picture with a video camera, binfile keyword, import command

To import a picture file into an image, you put the file's URL into the image. The following statement displays a JPEG file in an image:

```
put URL "binfile:mypict.jpg" into image "My Image"
```

Because a picture file consists of binary data, you must use the binfile URL scheme, not the file URL scheme.

Tip: To create a new image, use the import command, or choose File menu>Import As Control>Image File.

How to import a text file into a field

See Also:

How to export data to a file, How to import and export styled text, How to import an RTF file, How to import a Unicode text file, How to import data from a file, File menu > Import As Control > Text File..., file keyword, open file keyword, read from file keyword, URL keyword

You import the contents of a text file by choosing File menu>Import As Control>Text File, or by creating a field and using the put command with the fileÆs URL to import the text.

For example, to put the contents of a text file into a new field, use a set of statements like the following in a handler or the message box:

```
create field "Imported Text"  
put URL "file:/drive/data.txt" into field "Imported Text"
```

Tip: To work with a file too large to fit into available memory, use the read from file command. instead of using the put command with the fileÆs URL.

How to import a Unicode text file

See Also:

How to find out whether text in a field is Unicode, How to import and export styled text, How to import a text file into a field, How to import data from a file, unicodeText property, URL keyword

You use the unicodeText property to import a file that contains Unicode text. To put the text from a Unicode file into a field, use a statement like the following in a handler or the message box:

```
set the unicodeText of field "Text" to URL "binfile:my.txt"
```

If the file contains text in multiple languages, Revolution automatically sets the textFont of language runs to the appropriate Unicode font.

Important! This method works only if the file you are importing contains Unicode (UTF-16) data. It will not work for other encoding methods such as UTF-8 or Shift-JIS.

How to import an RTF file

See Also:

How to export text from a field in RTF format, How to import a text file into a field, How to import a Unicode text file, How to import and export styled text, file keyword, htmlText property, RTFText property, URL keyword

You use the RTFText property to import a file that contains styled text in RTF format. To put the text from an RTF file into a field, use a statement like the following in a handler or the message box:

```
set the RTFText of field "Text" to URL "file:test.rtf"
```

Important! The RTFText property supports only character styling (fonts, sizes, styles, and colors of text). A complex RTF file may contain footnotes, margin settings, and other information that cannot be displayed in a Revolution field. This information is ignored when you set the field's RTFText property.

How to import and export styled text

See Also:

How to copy styled text between fields, How to export text from a field in RTF format, How to import a Unicode text file, How to import an RTF file, How to store styled text in a variable or property, `htmlText` property, `RTFText` property

The easiest way to import and export styled text—that is, text that includes font, size, style, and color information—is to use the `htmlText` property. Many word-processing applications can read and write HTML files, and can understand the HTML produced by Revolution.

The following statement exports the text in a field—complete with styles—to an HTML file:

```
put the htmlText of field 1 into URL "file:myfile.html"
```

This statement imports an HTML file into a field:

```
set the htmlText of field 1 to URL "file:myfile.html"
```

How to import data from a file

See Also:

How to append data to the end of a file, How to display the contents of a text file, How to export data to a file, How to import a text file into a field, About filename specifications and file paths, About using URLs, uploading, and downloading, binfile keyword, file keyword, put command, read from file command, resfile keyword, URL keyword

You import data from a file using its URL. You can import data from files on your system (using the file or binfile URL scheme) or from files on the Web (using the ftp or http URL scheme).

For example, to get the contents of a text file on your drive, use a statement like the following in a handler or the message box:

```
get URL "file:/mydrive/test/data.txt"
```

Tip: To import binary (non-text) data from a file on your system, use the binfile URL type instead of the file URL type.

Tip: To work with a file too large to fit into available memory, use the read from file command.

How to import Shift-JIS Japanese text

See Also:

How to convert between Unicode and ASCII text, How to enter or display Unicode text in a field, How to find out whether text in a field is Unicode, How to import a Unicode text file, Why is there a problem with line endings?, textFont property, uniDecode function, uniEncode function

Revolution cannot display Shift-JIS text directly. However, it can convert Shift-JIS to Unicode and display the Unicode text. You use the uniEncode function to convert Shift-JIS text to Japanese Unicode text.

To import a text file containing Shift-JIS text into a field, use a set of statements like the following:

```
set the textFont of field "My Text" to ",Japanese"  
put URL "file:myJIS.txt" into inputText  
put uniEncode(inputText,"Japanese") into field "My Text"
```

The file `myJIS.txt` is converted to Unicode and displayed in the field `My Text`, using the default Japanese font in your system settings.

How to include a comma in a parameter

See Also:

comma constant

You include a comma in a parameter by either enclosing it in double quotes, or using the comma constant.

Since commas are used to separate parameters in Transcript statements, you can't include one in a parameter. For example, suppose you want to use a point (which consists of two numbers separated by a comma) as a parameter to a handler. The following statement sends the point as two parameters, not one:

```
myHandler 22,40
```

Quoting the parameter avoids this problem:

```
myHandler "22,40"
```

How to include a field on each page of a printed report

See Also:

To display a field in a printed report, you create a report viewer object on the report layout card, then link it to the field you want to display. Normally, a report viewer displays different contents for each card in the report, because the field's content is different on each card. To include the field contents from a particular card on all pages of the report, you lock the report viewer that's linked to the field:

1. Create a report viewer and link it to the field.
2. On the Basic Properties pane of the report viewer's property inspector, enter the number of the card you want to use in the "Page" box.
3. Click the Lock () button. to lock the report viewer to the specified card.

The report viewer displays the field's contents from that card on all pages of the report.

How to include a group on a card

See Also:

How to automatically include groups on a new card, About groups and backgrounds, Object menu > Place Group, place command, remove command

You include an existing group of objects on a card by choosing Objects menu>Place Group and choosing the group's name, or by using the place command in a handler or the message box.

Note: You can place a group on a card only once. If the group is already on the current card, its name does not appear in the Place Group submenu.

How to include a quote in an expression

See Also:

Tip: Easily use strings that contain quotes, & operator, && operator, quote constant

You include a double quote in an expression by using the quote constant.

Since double quotes are used to enclose literal strings in Transcript expressions, you can't include one in a literal string. For example, suppose you want to display a dialog box with this message: Say "Cheese" now! The following statement causes an error message, because Revolution interprets the second double quote before `öCheeseö` as the end of the parameter:

```
answer "Say "Cheese" now!" -- WON'T WORK
```

Replacing double quotes with the quote constant avoids this problem:

```
answer ("Say" && quote & "Cheese" & quote & "now!")
```

How to include a slash in a file path

See Also:

How to determine whether a file path is relative or absolute, How to include a comma in a parameter, How to include a quote in an expression, How to list the contents of a folder, Why can't Revolution find a file I specified?, revMacFromUnixPath function, revUnixFromMacPath function

To include a slash in a file path, substitute a colon (:) for the slash. For example, if you have a file named "And/Or", in a file path, write it like this: "/MyDisk/MyFolder/And:Or".

Revolution file paths are always written with a slash (/) separating the component levels. For example, if a file "MyFile.txt" is contained in the folder "MyFolder" which is found on the volume "MyDisk", its file path is written like this: "/MyDisk/MyFolder/MyFile.txt". On Unix and Windows systems, file and folder names cannot contain slashes.

However, a file or folder name that contains a slash on a Mac OS and OS X system can cause ambiguity, because Revolution can't tell whether the slash is part of the name or separates another folder level. Since the colon is the Mac OS path delimiter, it never appears in a Mac OS file or folder name, so this substitution is unambiguous.

How to include an image in a text field

See Also:

How to determine the dimensions of an image, How to display an icon or picture in a button, How to put text into a field, icon property, imageSource property

You use the imageSource property to include an image in a field. When you set the imageSource of a character in a field to an image reference, the image replaces the character.

The image reference you use can be a name or ID of an image in any stack thatÆs loaded into memory, or the URL of a picture in any format Revolution supports:

```
set the imageSource of char 22 of me to 3455
set the imageSource of char myChar of field "This Field"

to "My Image"
set the imageSource of char 456 of field 1

to "http://www.example.com/logo.gif"
```

How to include an object in the message path of every object

See Also:

How to call a custom function thatÆs not in the message path, How to intercept a message, How to trap a message before a handler is executed, How to use a handler outside the message path, About messages and the message path, insert script command, start using command

When creating an application, you may find that you want to use certain utility handlers often, from the scripts of many different objects. To make using the handlers more convenient, you can put them all in the script of an object, and place that object in the message path of every object so that any object can use the handlers.

You use the insert script command to place an object in the message path of every object. The following statement places a button in the message path, so its handlers can be used by any object:

```
insert the script of button "Library" into back
```

You can insert the script of any object thatÆs in a stack loaded into memory.

How to include comments and notes in a script

See Also:

How to temporarily remove a portion of a script, About the structure of a script, Script menu > Comment (Script Editor), Script menu > Uncomment (Script Editor), View menu > Default Comment Character... (Script Editor), `/**/` keyword, `--` keyword

Comments are ignored by Revolution, even if they contain Transcript statements. This means that you can place remarks and notes to yourself in a script, without confusing Revolution by trying to compile or execute these notes as though they were Transcript code.

Any line in a script (or portion of a line) that starts with two dashes (`--`) or a hash mark (`#`) is a comment.:

```
put 1 into myVar -- this part is a comment
```

You can make a multiple-line comment (called a block comment) by surrounding it with `/*` and `*/`:

```
answer "Hello!" /* This is a multiple-line comment that  
starts on the previous line and ends on this line */
```

How to install a plugin

See Also:

How to add custom utilities to the development environment, How to create a code library,
Development menu > Plugins, Development menu > Plugins > Plugin Settings

A plugin is simply a stack with whatever code and controls are needed to implement a utility you want to add to the development environment. You can create your own useful plugins, as well as downloading plugins from other developersÆ Revolution-related web sites.

To install a plugin in your copy of Revolution, you place the stack file in the ôpluginsö folder inside the Revolution folder. The next time you start up Revolution, you can choose the pluginÆs name from Development menu>Plugins to open the stack.

You use the Plugin Settings window to control the behavior of the plugin: what kind of window it appears in, whether it should open automatically, and which messages it should intercept. To open the Plugin Settings window, choose Development menu>Plugins>Plugin Settings.

How to intercept a message

See Also:

How to include an object in the message path of every object, How to trap a message before a handler is executed, About messages and the message path, insert script command, start using command

When creating an application, you may find that you want to trap certain messages before they reach the target object. (For example, you might want to prevent mouseUp handlers from being executed if the application is in a certain mode.)

You use the insert script command to place an object in the message path of every object. The following statement places a card at the beginning in the message path, so it can intercept any message sent to any object:

insert the script of card "Help" into front

If you place (for example) a mouseUp handler in the card's script, mouse clicks send the mouseUp message to the card before it's received by the object that was clicked.

How to investigate code in the development environment's windows

See Also:

How to add custom utilities to the development environment, How to get information about items on the screen, Edit menu > Preferences, Tools menu > Message Box, View menu > Revolution UI Elements in Lists, Shortcut to display a contextual menu

The Revolution development environment is a Revolution application, and the code that controls windows, buttons, and other controls is written in Transcript. You are welcome to examine and alter it for your own use.

To enable the ability to examine Revolution windows, choose Edit menu>Preferences. In the Shortcuts pane, check the box labeled Contextual menus work in Revolution windows. To examine a Revolution window, hold the mouse over the window you want to examine and use the contextual menu shortcut (Command-Control-Shift-click or Control-Shift-Right-click), then choose Edit Script to examine the script of the object you clicked.

Important! Runtime Revolution permits you to make changes to your own copy of the development environment, but does not provide support for them. You're on your own here.

How to investigate the Revolution custom libraries

See Also:

How to investigate code in the development environmentÆs windows, Animation Tutorial, Tools menu
> Message Box

The Revolution development environment is a Revolution application, and most of the custom libraries are written in Transcript. You are welcome to read this code.

To view the main Revolution libraries, click the `Back Scripts` button in the message box, and make sure the `Show Revolution IU` box at the bottom is checked. Select the library you want to examine and click `Edit Script` to see the Transcript code for the functions and commands in that library.

Important! Runtime Revolution permits you to modify the custom libraries for your own needs, but does not provide support for such changes. YouÆre on your own here.

How to keep a window on top of other windows

See Also:

How to bring a window to the front, How to hide other applicationsÆ windows, palette command, raisePalettes property, style property

A palette window floats in front of other windows, as long as the raisePalettes property is set to true. You might want to use a palette for any stack that must remain available and shouldnÆt be hidden behind other windows—for example, a tool bar.

To display a stack as a palette, you use the palette command:

```
palette stack "My Tools"
```

The palette command opens the stack in a palette window. If the stack is already open, its window is changed to a palette.

Note: If the style of the stack is set to anything other than `topLevel`, the palette command will not change its style. In this case, set the style to `palette` instead of using the palette command.

How to leave room on the screen for operating system elements

See Also:

To set aside space on the screen for elements such as toolbars and icons—whether they are part of your application or part of the operating system—you set the `windowBoundingRect` property.

The `windowBoundingRect` limits the area in which stack windows open, and specifies the maximum space in which they zoom or maximize. (The user can still manually move or resize a window so it extends outside the `windowBoundingRect` and into the reserved area.)

Tip: The initial setting of the `windowBoundingRect` property leaves room at the top of the screen for the menu bar (on Mac OS and OS X systems) and at the bottom of the screen for the task bar (on Windows systems).

How to let an object delete itself

See Also:

To delete an object, you use the delete command. However, you cannot use the delete command in an object's own script, because an object that contains a running handler can't be deleted until the handler is finished. To allow an object to delete itself, place a handler like the following in the script of an object that's further along the message path:

```
on deleteMe  
  delete the target  
end deleteMe
```

In the object's own script, call the `deleteMe` handler using the send...in time form of the send command (the delay ensures that the handler is finished when the delete command is executed):

```
send "deleteMe" to me in 10 milliseconds
```

How to list all open windows

See Also:

How to bring a window to the front, How to find out whether a window is open, Recipe for a Window menu, Recipe for finding out whether a window is open by title, label property, name property, openStacks function

You use the openStacks function to get a list of open stack windows:

put the openStacks into field "Window List"

Note: The openStacks function returns the name of stack windows, not their label (which is what is shown in the window's title bar). To find out whether there is an open window with a certain title, you need to check the label property of each stack returned by the openStacks.

Tip: In the Revolution development environment, open windows are listed in the Window menu. Choose a window to bring it to the front.

How to list an object's custom properties

See Also:

How to create a custom property, How to find out whether a custom property exists, About custom properties and custom property sets, Why doesn't a custom property appear in the property inspector?, customKeys property, customProperties property

You use the customKeys property of an object to find out what custom properties the object has. You can use the Custom Properties pane in the object's property inspector to see a list of its custom properties, or use the customKeys property in a handler or the message box.

Important! If the object has more than one custom property set, you must set the customPropertySet to the correct set before checking the customKeys.

The following example lists all the custom properties in a graphic:

```
put the customKeys of graphic 1 into myCustomProps  
answer myCustomProps
```

How to list the audio clips and video clips in a stack

See Also:

You use a repeat control structure to list the audio clips or video clips in a stack. The following example places the names of all audio clips in a variable:

```
repeat with x = 1 to the number of audioClips
  put the name of audioClip x & return after myVariable
end repeat
```

You can also see a list of audio clips and video clips in the Application Browser. First choose Edit menu > Preferences. In the "Application Browser" pane, check the boxes labeled "Show audio clips" and "Show video clips". Then choose Tools menu > Application Browser. When you click the disclosure triangle for a stack, the stack's audio clips and video clips are listed below the stack's name.

How to list the cards in a stack

See Also:

How to list the objects on a card, How to scan the cards in a stack, Independent Study Tutorial, Recipe for a Cards menu, Tools menu > Application Browser, cardNames property, cardIDs property

You can see a list of the cards in a stack in the Application Browser, or by using the cardNames property in a handler or the message box.

To see a list of cards in a stack using the Application Browser:, click the disclosure button next to the stackÆs name.

The following statement puts a list of cards into the it variable:

get the cardNames of this stack

How to list the contents of a folder

See Also:

How to change the current folder used for file operations, How to list the files in an FTP directory, About filename specifications and file paths, Why canÆt Revolution find a file I specified?, answer folder command, defaultFolder property, files function, folders function

You get the contents of a folder from within a handler using the files and folders functions.

The files function returns a list of files in the defaultFolder, while the folders function returns a list of folders in the defaultFolder. The combination of these lists is the contents of the defaultFolder.

First, set the defaultFolder to the pathname of the folder whose contents you want to list. Then use the files and folders functions to get the list:

```
set the defaultFolder to "/Hard Disk/My Folder/"  
put the files & return & the folders into myContents
```

How to list the files in an FTP directory

See Also:

How to create a directory on an FTP server, How to download a file from an FTP server, How to list the contents of a folder, How to remove a file from an FTP server, How to upload a file to an FTP server, How to use a password for an FTP server, About using URLs, uploading, and downloading, ftp keyword, libURLftpCommand function, libURLSetFTPListCommand command

You use an ftp URL to list the files in an FTP directory.

A URL that ends with a slash (/) designates a directory (rather than a file). An ftp URL to a directory evaluates to a listing of the directory's contents (its files and subdirectories). The following statement gets the contents of an FTP directory, and places the list in a field:

```
put URL "ftp://ftp.example.net/mydir/" into field "List"
```

Note: By default, the list includes information such as permissions, owner, size, and last modification date, as well as the name of each file or subdirectory. You can switch to a simpler format using the libURLSetFTPListCommand command.

How to list the objects on a card

See Also:

How to list the cards in a stack, Tools menu > Application Browser, control keyword, mouseControl function, number function

You can see a list of the objects on a card using the Application Browser:

1. Find the stack in the list, and click the disclosure button next to it to reveal a list of all cards in that stack.
2. Click the card's name to select it.

The objects on the card are displayed on the right side of the Application Browser.

How to load a stack into memory

See Also:

How to open a stack, About main stacks, substacks, and the organization of a stack file, `destroyStack` property, `stackFiles` property

The objects in any open stack are accessible to other stacks. You can also make the contents of a stack accessible without opening it, by loading it into memory. To load a stack into memory, get one of the `stackÆs` properties, referring to the stack by its filename:

```
get the name of stack "/Disk/Folder/stack.rev"
```

This loads the main stack in the file. The stack (and any substacks) will remain loaded into memory until you use the `delete stack` command to remove it from memory.

Important! If one stack in a stack file is loaded into memory, so are any other stacks in the same stack file. You cannot load one stack in a stack file without loading all the rest at the same time.

How to make a system call

See Also:

How to change behavior depending on the platform, do command, open process command, platform function, read from process command, setRegistry function, shell function, systemVersion function, write to process command

Transcript does not allow making direct calls to the operating system. There are a number of ways to access OS features, depending on the functionality you need and how the operating system allows access to it:

- ò On Mac OS and OS X systems, you can use the do as applescript form of the do command to execute AppleScript code.
- ò On Windows systems, you can use the setRegistry function to set an entry in the Windows registry.
- ò On Unix, OS X, and Windows systems, you can use the shell function or open process command to call a shell command and, if necessary, exchange data with it.
- ò On any operating system, you can write an external in C or another compiled language to interface with the operating system.

How to make a throbbing button

See Also:

How to get a striped background in OS X, How to make the Return or Enter key activate a button, default property

On OS X systems, the default button in a dialog box throbs rhythmically to catch the user's attention.

To make a button throb when displayed on an OS X system, set the button's default property to true. You can either set this property in a handler or the message box, or use the Basic Properties pane in the button's property inspector to check the "Default Button" option.

Important! For default buttons to throb, the application must be running on an OS X system, and the lookAndFeel property must be set to "Appearance Manager".

How to make an alias, symbolic link, or shortcut to a file

See Also:

How to create a file, About filename specifications and file paths, `alias` Reference function, `create alias` command

You use the `create alias` command to make an alias (Mac OS or OS X), symbolic link (Unix), or shortcut (Windows). The terminology varies depending on platform.

To create an alias, you specify the path to the new alias and to the file:

```
create alias "/Users/me/My File"
```

```
to file "/Volumes/Current Stuff/My File"
```

Important! When creating an alias on Windows systems, make sure the alias's name ends with the extension `.lnk` to ensure that it behaves properly when the user double-clicks it.

How to make an object the same color as its owner

See Also:

How to change an object's color, How to find out the actual color of an object, How to restore the default colors of an object, About colors and color references, backgroundColor property, borderColor property, bottomColor property, colors property, effective keyword, focusColor property, foregroundColor property, hiliteColor property, owner property, shadowColor property, topColor property

You let an object inherit a color from its owner by setting the relevant color property to empty. For example, to let a field inherit the foregroundColor of the group the field is in, use a statement like the following:

set the foregroundColor of field 1 of group "Stuff" to empty

You change the colors of an object by setting the object's eight color properties. If a color property of the object is empty—that is, if it's not set to a color—the owning object's color shows through.

Tip: To set all an object's colors at once, set the object's colors property. Setting the colors property sets all eight of the color properties (backgroundColor, foregroundColor, borderColor, topColor, bottomColor, hiliteColor, shadowColor, and focusColor).

How to make changes to a SQL database

See Also:

How to save changes to a SQL database, About connecting to and using SQL databases, revCommitDatabase command, revExecuteSQL command, revRollBackDatabase command

You use the revExecuteSQL command with the appropriate SQL statements to make changes to a database. (To specify the database, you use the ID number returned by the revOpenDatabase function.)

This example creates a new table with two columns in the database whose ID number is 12:

```
revExecuteSQL 12,"CREATE TABLE NewOrders (OrderID,Item)"
```

Other operations, such as inserting and deleting records or changing the value in a database field, are performed similarly, using the appropriate SQL statements.

How to make subscripts and superscripts

See Also:

How to change the size of text, How to change the style of text, Text menu > Subscript, Text menu > Superscript, fixedLineHeight property, textShift property, textSize property

You set the textShift property to make text in a field subscripted or superscripted. The following example makes a subscript:

set the textShift of char 12 of field 1 to 3 -- shift down

ò To make text into a subscript, set its textShift to an integer greater than zero. The text is shifted down by the specified number of pixels.

ò To make text into a superscript, set its textShift to an integer less than zero. The text is shifted up by the specified number of pixels.

ò To make the text normal again, set its textShift to zero.

To superscript or subscript the selected text (and also make the text smaller), choose Text menu>Subscript or Text menu>Superscript.

How to make the computer speak out loud

See Also:

How to beep, How to speak several phrases in succession, Recipe for speaking an alert message, `revIsSpeaking` function, `revSetSpeechPitch` command, `revSetSpeechSpeed` command, `revSetSpeechVoice` command, `revSpeech` command

You cause the computer system to speak text by using the `revSpeak` command. This command uses the text-to-speech capabilities of the operating system to speak a text phrase. To speak a phrase, use a statement like the following:

```
revSpeak "Hello there!"
```

You can speak a literal string or the contents of a container (such as a field or variable).

Note: Text to speech is not supported on Unix systems.

Tip: Use the `revSetSpeechPitch`, `revSetSpeechVoice`, and `revSetSpeechSpeed` commands to change the way the computer voice sounds.

How to make the Return or Enter key activate a button

See Also:

How to assign a keyboard equivalent to a button, How to make a throbbing button, How to simulate the action of a button, default property, enterInField message, enterKey message, returnInField message, returnKey message

In most dialog boxes, you click a button to dismiss the dialog and perform the action of the dialog box. Standard user interface guidelines call for the most usual (or safest) button to be chosen automatically when the user presses Enter or Return.

To specify that a button will be chosen when the user presses Enter or Return, set the button's default property to true.

Tip: If the dialog box contains editable fields, include an enterInField and returnInField handler in the script of the field, card, or stack. These handlers should send a mouseUp message to the default button. This ensures that the Enter or Return key will activate the button even if there is an insertion point or text selection.

How to minimize a window with a custom shape

See Also:

A stack whose windowShape property is set to true does not have an minimize box or collapse box. To provide this feature, create a button or other control that sets the stack's iconic property to true:

```
on mouseUp -- might be in an "iconify" button
    set the windowShape of this stack to zero -- normal shape
    set the iconic of this stack to true
end mouseUp
```

To restore the custom window shape when the stack is un-minimized, include a handler like this in the stack script:

```
on uniconifyStack
    set the windowShape of this stack to 92773 -- the ID
end uniconifyStack
```

How to monitor messages as they're sent

See Also:

How to enter the debugger, How to monitor the value of variables while debugging, About messages and the message path, Development menu > Message Watcher, Debug menu > Message Watcher (Script Editor)

You use the message watcher to view messages being sent (along with function calls, getProp calls, and setProp triggers). To display the message watcher, choose Development menu>Message Watcher.

As each message is sent, its name appears in the message watcher. To find out a message's target, click the message name in the scrolling list. The name of the target object appears at the bottom of the message watcher.

To prevent certain messages from appearing—for example, if you want to track a particular kind of message and ignore all others—click Suppress. In the dialog box, check the boxes for the messages you don't want to see.

Note: You can also open the message watcher in the debugger, by choosing Debug menu>Message Watcher.

How to monitor the value of variables while debugging

See Also:

How to enter the debugger, How to monitor messages as they're sent, About containers, variables, and sources of value, Development menu > Variable Watcher, Debug menu > Variable Watcher (Script Editor), variableNames function

You use the variable watcher to monitor the value of variables. To display the variable watcher, choose Debug menu>Variable Watcher. When the debugger is running, the variable watcher shows all currently defined variables along with their values.

Tip: To automatically stop the debugger when a variable changes value, click to the left of the variable's name in the variable watcher. To stop when the variable equals an expression, enter the expression in the dialog box and click OK. To stop whenever the variable changes, click Cancel instead.

You can also open the variable watcher when the debugger is not running, by choosing Development menu>Variable Watcher. If the debugger is not running, the variable watcher shows global variables.

How to move a card between stacks

See Also:

How to copy a card, How to move a card within a stack, How to select a card, Edit menu > Cut, Edit menu > Copy, Edit menu > Paste, copy command, cut command, paste command

You move a card to another stack using cut (or copy) and paste.

To move a card from one stack to another, select the card, then choose Edit menu>Cut (to remove the card from the stack) or Edit menu>Copy (to leave a copy in the original stack). Then go to the stack where you want to place the card and choose Edit menu>Paste. The pasted card appears after the card you were on when you chose the menu item.

You can also use the cut or copy and paste commands in a handler or the message box. To move a card to another stack with a single command, use the copy command:

```
copy card "Some Card" to stack "Some Stack"
```


How to move a card within a stack

See Also:

How to move a card between stacks, number property

You move a card within a stack by changing its number property to the desired position. For example, to move a card to be the first card in its stack, use a statement like the following:

set the number of card "Contents" to 1

To move a card using the property inspector, choose "Size & Position" from the menu at the top of the inspector, then change the number in the Layer box to the new card position.

How to move a referenced image into a stack

See Also:

A referenced image is stored in a separate file and displayed in the image object. To import an image into a stack, you put the file's URL into the image.

The filename property of a referenced image consists of the path to the picture file it displays. The following statement moves a referenced image's file into the image:

```
put URL ("binfile:" & the filename of image "My Image")  
  
into image "My Image"
```

(Because a picture file consists of binary data, you must use the binfile URL scheme, not the file URL scheme.)

How to move a stack to another file

See Also:

File menu > Save As..., File menu > Move Substack to File..., Tools menu > Application Browser, mainStack property

You move stacks between files by setting the mainStack property of the stack you're moving to the main stack name of the destination stack file.

To move a stack to another file, set its mainStack property to the name of the main stack of the destination stack file. For example, if you want to move a stack into a file whose main stack is named "My Main", set the stack's mainStack to "My Main". This makes your stack a substack of "My Main". The next time you save the stack file, your stack is saved along with it in the same file.

You can either set this property in a handler or the message box, or use the "Main Stack" menu in the Basic Properties pane of the stack's property inspector.

How to move cards from one stack to another

See Also:

Edit menu > Paste, Object menu > New Card, copy command, cut command, paste command

You move cards from one stack to another by using the copy command in a handler or the message box, then using the delete command to remove the old cards.

Using the copy command, you can specify which stack to copy the card to. You can also use a repeat control structure to copy multiple cards.

If you copy a card without specifying a destination stack, the card is placed on the clipboard. You can open the stack where you want to move it, then choose Edit menu>Paste to paste it into the current stack. A pasted card is placed immediately after the current card in the stack.

How to move one control in front of another

See Also:

How to bring a control to the front, How to change the order of cards in a stack, How to change the tab order of controls on a card, Why doesn't the Tab key move to the next field?, Object menu > Move Forward, Object menu > Move Backward, layer property, number property

You move one control in front of another by setting the control's layer property.

The layer of a control determines its back-to-front order on the card. For example, the bottommost control's layer is 1. If two controls overlap, the one with the higher layer is in front.

You can either set this property in a handler or the message box, or use the Size & Position pane in the card's property inspector to change the control's layer. You can also select the control and choose Object menu>Move Forward to move the control forward one layer.

How to move through the records in a record set (database cursor)

See Also:

How to create a record set in a SQL database, How to find out which record sets are open in a database, How to navigate among records from an automatic database query, How to use a SQL query to select records in a database, About connecting to and using SQL databases, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, revNumberOfRecords function

When you use the revQueryDatabase or revQueryDatabaseBLOB command, you create a record set (database cursor) containing a subset of the records in the database. You move among the records using the revMoveToFirstRecord, revMoveToLastRecord, revMoveToNextRecord, and revMoveToPreviousRecord commands.

This example moves to the twelfth record in the record set whose ID number (returned by revQueryDatabase or revQueryDatabaseBLOB) is in the variable `thisCursor`:

```
revMoveToFirstRecord thisCursor -- start at the beginning
repeat for 12 times
  revMoveToNextRecord thisCursor
end repeat
```

How to navigate among records from an automatic database query

See Also:

How to display a single record from an automatic database query, How to move through the records in a record set (database cursor), How to save changes to records from an automatic database query, How to set up an automatic database query, About connecting to and using SQL databases, `revMoveToFirstRecord` command, `revMoveToLastRecord` command, `revMoveToNextRecord` command, `revMoveToPreviousRecord` command

You use the Database pane in a button's Properties palette to navigate among the records fetched by an automatic database query. Follow these steps:

1. Create or select a button and choose Object menu>Object Inspector, then choose "Database" from the menu at the top of the inspector.
2. From the Query menu, choose the automatic query you want to use.
3. From the Action menu, choose "Move to previous record" or "Move to next record".

When you click the button, any objects on the current card that display data from the same automatic query now display the data from the previous or next record.

How to open a stack automatically when starting Revolution

See Also:

How to add custom utilities to the development environment, How to install a plugin, How to open a stack, Development menu > Plugins > Plugin Settings, startup message

To open a particular stack automatically when starting Revolution, make it into a plugin.

A plugin is simply a stack with whatever code and controls are needed to implement your utility. You place the stack file in the `plugins` folder inside the Revolution folder. The next time you start up Revolution, follow these steps:

1. Choose Development menu>Plugins>Plugin Settings.
2. Choose your stack's name from the Plugin menu at the top of the Plugin Settings window.
3. Choose the `Revolution starts up` option.

How to open a stack

See Also:

How to bring a window to the front, How to display another stack in the current window, How to go to another card, How to respond when a stack opens, How to use a stack thatÆs on a web server, Why canÆt I open a downloaded stack?, Why does Revolution ask to purge a stack?, File menu > Open Stack..., File menu > Open Recent Stack, Tools menu > Application Browser, go command, modal command, modeless command, palette command, topLevel command

You open a stack by choosing File menu>Open Stack, or by using the go command in a handler or the message box.

Using the go command, you can specify any stack, or use a file path obtained from the user with the answer file command.

Tip: To open a stack as a palette, modal dialog box, or modeless dialog box, use the palette, modal, or modeless commands instead of the go command.

How to open a URL in a browser

See Also:

How to display a web page in a field, How to fetch data from the Internet, How to use a stack thatÆs on a web server, http keyword, revGoURL command, revMail command

You use the revGoURL command to open a URL in the userÆs preferred browser. The following statement launches the browser (if necessary) and opens a web page in it:

```
revGoURL "http://www.example.com/index.html"
```

Tip: On Mac OS and OS X systems, you can use the revGoURL command to access any type of URL, in the application thatÆs set up to handle that URL scheme in the Internet control panel:

```
revGoURL "ftp://ftp.example.org/incoming/" -- FTP program  
revGoURL "mailto:bob@example.com" -- email program
```

How to open and close a drawer

See Also:

On OS X systems, a drawer is a subwindow that slides out from one edge of a parent window. Drawers are usually used to allow access to settings that should be quickly accessible, but don't need to be visible all the time.

In Revolution applications, drawers are implemented as stacks. First create the stack you want to make into a drawer, with all its controls. Once you've created the stack, you use the drawer command to display it:

```
drawer stack "Tools" in stack "Main"
```

To slide the drawer back in, simply close the stack:

```
close stack "Tools"
```

How to peek at invisible objects in a stack

See Also:

Why do icons disappear from a standalone application?, View menu > Show Invisible Objects, hide command, show command, showInvisibles property, visible property

You display invisible objects by setting the showInvisibles property to true. You can either set the showInvisibles in a handler, or choose View menu>Show Invisible Objects.

When the showInvisibles is true, you can see and select invisible objects, and the objects respond to mouse clicks.

To set the showInvisibles back to false, choose View menu>Show Invisible Objects again to uncheck the menu item.

How to pin an object to a window edge during window resizing

See Also:

You use the Geometry pane in an object's property inspector to keep the object a specified distance from one of the window's edges. In the object's property inspector, choose "Geometry" from the menu at the top, then follow these steps:

1. Click the "Position selected object" option. In the Geometry pane, you see a picture of a window, with a rectangle representing the object in the middle.
2. Click the gray bar leading from the object to the right or bottom edge of the window (or both), to turn the gray bar red.

When you resize the window, the object moves automatically to keep the same distance from the window edge.

How to place the insertion point in a field

See Also:

How to find out where the insertion point is, How to select text in a field, How to simulate a mouse click, after keyword, before keyword, select command, selection keyword

You use the select command to place the insertion point in a field.

If you use the select command with the before or after keyword, it places the insertion point at the location you specify. The following examples show how to place an insertion point at various points in a field's text:

- select before text of field "My Field" -- at start of text
- select after text of field ID 3452 -- at end of text
- select before line 3 of field 1 -- start of line 3
- select after char 22 of field 5 -- between char 22 and 23

How to play a streaming QuickTime file

See Also:

filename property, start command, URL keyword, Why don't movies play?, File menu > New Referenced Control > QuickTime-Supported File...

A streaming file is a video or sound file, located on an Internet server, which can begin playing before downloading is complete. To be capable of streaming, the file must be specially prepared on the server.

You play a streaming QuickTime movie by creating a player object and setting its filename property to the URL of the file. Then use the start command to start the movie:

```
set the filename of player "My Player"  
to "http://www.example.com/stream.mov"  
start player "My Player"
```

The file will automatically begin streaming into the player. It starts playing immediately, even while the rest of the file is being downloaded from the Web site to your system.

How to pretty print a script

See Also:

How to automatically color-code a script, Script menu > Format (Script Editor)

Pretty printing is the use of indenting to show the structure of a handler. When pretty printed, the contents of control structures (such as if/then/else conditionals and repeat loops) are indented. Nested control structures are indented more, to show the level of nesting.

The Revolution script editor automatically pretty-prints the current handler when you place the insertion point somewhere in the handler and press Return or Tab. You can also choose Script menu>Format to pretty print the current handler.

To turn off automatic pretty printing, choose Edit menu>Preferences, then uncheck the box labeled "Auto-indent to show structure" in the "Script Editor" pane.

How to prevent a handler from executing

See Also:

Development menu > Suppress Messages, lock messages command, lockMessages property

You prevent a handler from executing by setting the lockMessages property to true. This prevents handlers such as mouseUp and openCard from executing by preventing messages from being sent.

You can either use the lock messages command to set this property in a handler, or choose Development menu>Suppress Messages so that the menu item is checked.

Tip: To go to a card or stack without seeing any actions it normally performs when opened, enter the following in the message box:

lock messages ; go to card "Card You Want to Go To"

How to prevent changing a checkbox's state by clicking it

See Also:

The state of a button such as a radio button or checkbox is specified by the button's `hilite` property. To prevent the user from changing this property when clicking the control, set the control's `autoHilite` property to `false`.

You can either set this property in a handler or the message box, or use the Basic Properties pane in the button's property inspector.

How to prevent changing a field's text

See Also:

How to allow copying text from a locked field, How to display text on a card, How to prevent entering certain characters in a field, How to respond to a change in field contents, How to select text in a field, Object menu > Object Inspector, lockText property, traversalOn property

You prevent the user from changing a field's text by setting the field's lockText property to true.

If the field's traversalOn property is also true, the user can select text and copy it, but not change it. If the traversalOn is false, the user cannot select text in the field.

If the lockText is true, you can still change the text by using the put command. The lockText property affects only typing into the field.

You can either set this property in a handler or the message box, or use the Basic Properties pane in the field's property inspector to lock or unlock the text.

How to prevent changing a stack

See Also:

How to prevent changing a field's text, Object menu > Stack Inspector, cantDelete property, cantModify property, lockText property

You prevent the user from changing a stack by setting the stack's cantModify property to true. When a stack's cantModify property is true, the user cannot move, resize, create, or delete objects, but can edit text in unlocked fields.

You can either set this property in a handler, or use the Basic Properties pane in the stack's property inspector to check the box labeled "Can't Modify".

Note: The cantModify property restricts user actions, but does not affect actions performed by a handler. To prevent handlers from deleting a card, group, or stack, use the cantDelete property.

How to prevent displaying intermediate changes on a card

See Also:

Why does an object or window flicker?, lock screen command, lockColorMap property, lockScreen property, unlock screen command

To prevent the user from seeing changes your scripts make, you use the lock screen command in your handler before making the changes, then use the unlock screen command to show the new appearance.

For example, suppose your handler hides one button and then shows another in its place, and you don't want the user to see a blank spot for a moment. To prevent this, use the lock screen command:

```
lock screen
hide button "Go"
show button "Stop"
unlock screen
```

How to prevent dragging and dropping to a field

See Also:

How to allow copying text from a locked field, How to copy dragged and dropped text, How to respond to a drag and drop, acceptDrop property, dragDrop message, dragEnd message, dragEnter message

You prevent dropping data into a field during a drag and drop by setting the acceptDrop property to false when the mouse pointer enters the field.

If the acceptDrop is set to false, when you drop data, no dragDrop message is sent to the field. Since the drop is automatically processed only when Revolution receives a dragDrop message, this prevents the usual automatic drop behavior.

Usually, you should set the acceptDrop in a dragEnter handler, as in the following example:

```
on dragEnter -- in a field script
  set the acceptDrop to false
end dragEnter
```

How to prevent entering certain characters in a field

See Also:

How to prevent changing a field's text, How to respond to a change in field contents, Recipe for limiting the number of characters in a field, Recipe for rejecting non-digit characters typed into a field, `keyDown` message, `rawKeyDown` message

You use a `keyDown` or `rawKeyDown` handler to prevent certain characters from being typed into a field.

The handler intercepts each keystroke, and the character is entered into the field only if you pass the `keyDown` or `rawKeyDown` message.

How to prevent interrupting a handler

See Also:

How to stop a running handler, Why can't I interrupt a handler?, Shortcut to stop a running handler, allowInterrupts property, cantAbort property, errorDialog message, interrupt function, lockErrorDialogs property

The user can stop a running handler by pressing Control-period or Control-break (on Windows or Unix) or Command-period (on Mac OS or OS X). Certain handlers should not be interrupted. For example, handlers that modify system files can leave the operating system in an unworkable state if they're interrupted in the middle. In such cases, the user should be prevented from interrupting a handler during the critical portions of code.

You prevent the user from interrupting a handler by setting the allowInterrupts property to false. (Be sure to set the property back to true when it is safe to do so.)

Tip: You can call the interrupt function to check whether the user has tried to interrupt the handler, so you can stop a lengthy process after doing any needed cleanup.

How to prevent the debugger from appearing

See Also:

How to enter the debugger, Development menu > Script Debug Mode, Debug menu > Script Debug Mode (Script Editor)

To prevent the debugger from appearing when a breakpoint is encountered, choose Development menu>Script Debug Mode to uncheck the menu item. (In the script editor, you can choose Debug menu>Script Debug Mode. Both menu items control the same setting, and when you uncheck one, the other is also unchecked.)

When script debugging is turned off, the development environment ignores breakpoints and does not offer a Debug button in the error window. Use this feature to temporarily ignore breakpoints in order to test a script without running the debugger.

To allow use of the debugger, choose Development menu>Script Debug Mode (or, in the script editor, Debug menu>Script Debug Mode) again to check the menu item.

How to prevent the user from moving a window

See Also:

How to prevent the user from resizing a window, How to remove a window's title bar, How to respond to moving a window, About windows, palettes, and dialog boxes, decorations property, title keyword

You prevent the user from moving a stack window by removing the stack's title bar.

To remove the title bar (and any draggable borders), set the stack's decorations property to empty. You can set this property in a handler or the message box, or in the stack's property inspector.

How to prevent the user from resizing a window

See Also:

How to change the size and position of a window, How to prevent the user from moving a window, About windows, palettes, and dialog boxes, maxHeight property, maxWidth property, minHeight property, minWidth property, resizable property, resizeStack message

You prevent the user from moving a stack window by setting the stack's resizable property to false. You can set this property in a handler or the message box, or in the stack's property inspector.

Tip: To allow the user to change the stack window's height but not its width, set the stack's minWidth and maxWidth properties to the same number. To allow changing the width but not the height, set the stack's minHeight and maxHeight to the same number.

How to preview a printed report

See Also:

How to create and store report printing settings for later use, How to create a report layout card, How to print a report, How to specify what cards to include in a printed report, Tools menu > Report Builder, Object menu > New Control > Report Object, revPrintReport command

To preview what each page of a report will look like when it's printed, first go to the report layout card you want to use. Then choose View menu>Go to Report Page and choose one of the menu items in the submenu.

The settings you specified in the Report Builder determine which cards of which stack will be printed in the report. When you preview a page of the report, the fields on each card that will be printed appear in their linked report viewer objects, so you can see what each page will look like when the report is printed.

How to preview how a stack will look on other platforms

See Also:

How to change behavior depending on the platform, How to distribute a stack to users on other platforms, Supported Platforms Reference, lookAndFeel property, platform function

You preview how a stack will look on platforms other than the one you are developing on by setting the lookAndFeel property.

You can either set this property in a handler or the message box, or use the "Preview" submenu in the View menu to switch between the look and feel of Mac OS and OS X, Unix, and Windows.

Note: Changing the lookAndFeel property provides an approximation of appearance and behavior on other platforms. This approximation is not perfect; for example, the placement of the menu bar does not change according to the lookAndFeel.

How to print a card

See Also:

How to print selected cards as a single print job, How to print the contents of a field, Recipe for printing all cards that contain a word, Recipe for printing the fields on a card, File menu > Print Card..., answer printer command, print command

You print a card by choosing File menu>Print Card or by using the print command in a handler.

The `Print Card` menu item prints the entirety of the current card. If you use the print command, you can specify a card other than the current one, and print only a portion of the card.

On Mac OS, OS X, and Windows systems, the current printer is used. On Unix systems, `Print Card` creates a PostScript file, then runs the program specified by the `printCommand` property.

How to print a report

See Also:

How to create a custom printed report, How to create a report layout card, How to preview a printed report, How to specify what cards to include in a printed report, Tools menu > Report Builder, Object menu > New Control > Report Object, revPrintReport command

To print a report, follow these steps:

1. Create a card and lay out the report as you want it to appear, using report viewers and other objects.
2. Choose Tools menu>Report Builder and specify the settings for the report. (These settings are stored with the current card.)
3. To print the report immediately, click Print Report in the Report Builder window.

To print the report in a handler, first go to the report layout card you created, then use the revPrintReport command. The revPrintReport command uses the settings you created for the card in the Report Builder.

How to print all the cards in a stack

See Also:

How to print a card, How to print selected cards as a single print job, Recipe for fitting a printout to the page, File menu > Print Card..., open printing command, print command

You print all the cards in a stack by using the print command in a repeat loop, using the open printing command to print all the cards in a single print job:

```
open printing
repeat with x = 1 to the number of cards
  print card x
end repeat
close printing
```

If there is room, multiple cards are printed on the same page.

Tip: To print a stack other than the current stack, first set the defaultStack property to the name of the stack you want to print.

How to print in landscape mode

See Also:

answer printer command, open printing command, print command, printRotated property, revPrintField command, revPrintText command, revShowPrintDialog command

You specify printing in landscape mode with the printRotated property.

To print in landscape mode, set the printRotated to true before printing:

```
set the printRotated to true
print card "My Card"
```

Note: The printRotated property has no effect on Mac OS and OS X systems. To print in landscape mode on Mac OS or OS X, use the answer printer or revShowPrintDialog command to display the Page Setup dialog box before printing, then choose the Landscape option in the dialog box.

How to print selected cards as a single print job

See Also:

How to create a custom printed report, How to print a card, How to print the contents of a field, close printing command, open printing command, print command

You print a card by using the print card command. To group printouts into a single print job, you enclose all the print card commands with the open printing and close printing commands.

The following set of statements prints three cards:

```
open printing
print card "My Card"
print card "Another Card" of stack "Second Stack"
print card "Still Another Card"
close printing
```

When the close printing command is issued, all three cards are printed as a single print job.

How to print the contents of a field

See Also:

How to create a custom printed report, How to print a card, Recipe for printing the fields on a card, File menu > Print Field..., `htmlText` property, `revPrintField` command, `revPrintText` command

You print the text in a field by choosing File menu>Print Field, or by using the `revPrintField` command in the message box or a handler. All style information such as bold and italic, images embedded using the `imageSource` property, and so on, is included in the printout.

Tip: To print only the plain text of a field without styles, use the `revPrintText` command in a statement such as the following:

```
revPrintText (field "My Field")
```

(Be careful to include a space between `revPrintText` and the parenthesis.)

Tip: To print only a portion of a field while retaining the styles of text, use the `revPrintText` command in a statement such as the following:

```
revPrintText (the htmlText of line 1 to 10 of field 2)
```

How to put different-sized cards in the same stack

See Also:

How to change the size and position of a window, height property, location property, preOpenCard message, rectangle property, revChangeWindowSize command, width property

Individual cards in a stack do not really have a size; their display size is the same as the size of the stack window they're in. However, you can adjust the size of the stack to accommodate different cards with differing layouts.

You change the size of the stack window by changing its height and width properties or by using the revChangeWindowSize command. You can either predetermine the height and width for each type of card, or base the new height and width on the location of the card's objects.

Tip: If these changes are made in a preOpenCard handler, which is executed before the destination card is shown, the window is resized before the card's objects are shown. With this method there is no visible distortion during the movement from card to card: the window simply resizes itself.

How to put styled text on the clipboard

See Also:

How to allow copying text from a locked field, How to clear the clipboard, How to copy an image to the clipboard, How to copy dragged and dropped text, How to select text in a field, Why can't I select, copy, or paste text?, Recipe for collecting text selections on the clipboard, Edit menu > Copy, clipboard property, clipboardData property, HTMLText property, put command

You change the contents of the clipboard using the clipboardData property. To put styled text from a field on the clipboard, use the field's htmlText property along with the `HTML` element of the clipboardData:

set the clipboardData["HTML"] to the htmlText of field 1

You can use a similar statement to put a portion of a field on the clipboard:

set the clipboardData["HTML"] to

the htmlText of line 4 to 7 of field "My Field"

You can also create your own styled text, in the same format as the htmlText property:

set the clipboardData["HTML"] to "Bold!"

How to put text into a field

See Also:

How to prevent changing a field's text, How to respond to a change in field contents, How to select text in a field, Why can't I select, copy, or paste text?, put command, select command, traversalOn property

You use the put command to put text into a field. For example, to put the text "Hello World" into a field named "My Field", use a statement like the following (in a handler or in the message box):

```
put "Hello World" into field "My Field"
```

To put text at the beginning of a field without disturbing the field's contents, use a statement like the following:

```
put "text" before field "Field"
```

To put the text at the end of the field, use a statement like the following:

```
put "text" after field "Field"
```

How to quickly display the Transcript Dictionary

See Also:

About the Revolution documentation, Shortcut to open Revolution Documentation, Help menu > Transcript Dictionary

As you become more expert, you may find that you most often use the documentation to look something up in the Transcript Dictionary. You can make this process faster by setting the Transcript Language Dictionary as the first page to appear when you open the documentation window.

First open the documentation window and go to the page. Check the box labeled "Show this page when documentation opens" at the bottom of the window. Then close the window.

From now on, when you choose "Revolution Documentation" from the Help menu (or use one of the keyboard shortcuts), the Transcript Language Dictionary page appears automatically. You can open the documentation window, type the name of the term you want to look up, and press Return or Enter to open the Dictionary to that term—all without taking your hands off the keyboard.

How to quit the application

See Also:

How to close a window, How to respond to quitting an OS X application, About menus and the menu bar, quit command

You quit the application with the quit command.

Usually, you place a `Quit` menu item in the File menu, as the last menu item in the menu. The File menu's menuPick handler should contain the quit command:

```
case "Quit"  
    quit  
    break
```

Note: A Revolution application quits automatically when all its windows are closed and all waiting messages have been handled (including messages sent with the send...in time form of the send command).

How to refer to a control on another card

See Also:

You refer to a control thatÆs in another card or stack by including as many of the controlÆs owners as necessary to completely specify that particular control.

To refer to a control on a card other than the current card, specify the card along with the control, as in this example:

show button "My Button" of card 12

To refer to a control in another stack, specify the stack as well:

show button "My Button" of card 12 of stack "My Stack"

Note: If the object is in another stack, the stack must be open or loaded into memory, or included in an open stackÆs stackFiles property. Otherwise, you must specify the stack by its long name.

How to refer to a custom property in a non-active set

See Also:

How to create a custom property set, How to list an object's custom properties, About custom properties and custom property sets, Why doesn't a custom property appear in the property inspector?, customKeys property, customProperties property, customPropertySet property, customPropertySets property

If an object has more than one custom property set, you can refer to custom properties that aren't in the active set in one of two ways:

ò Switch to the custom property set you want, then refer to the custom property directly:

- set the customPropertySet of button ID 12 to "myProps"
- set the myCustomProp of button ID 12 to true

ò Use array notation to specify the custom property set you want, along with the custom property name:

- set the myProps["myCustomProp"] of button ID 12 to true

How to remove a file from an FTP server

See Also:

How to delete a file, How to download a file from an FTP server, How to upload a file to an FTP server, How to use a password for an FTP server, About using URLs, uploading, and downloading, delete URL command, ftp keyword, libURLftpCommand function

You use the delete URL command to remove a file from an FTP server. The following statement deletes a file named `deadfile.txt` from an FTP server, using the user name `me` and the password `pass`:

```
delete URL "ftp://me:pass@ftp.example.org/deadfile.txt"
```

Note: Normally, FTP servers do not allow anonymous users to delete files, for obvious reasons. This means that while an ftp URL without a user name and password is valid, you will almost always need a user name and password to use the delete URL command.

How to remove a single style from text

See Also:

How to change the style of text, How to remove all styles from text, Recipe for handling selection of choices from a Text menu, Shortcut to remove font changes from text, itemOffset function, textStyle property

You remove a style from text by either selecting the text and choosing the style from the Text menu to remove the checkmark next to the style, or by using the textStyle property in a handler.

The textStyle is a list of all styles that apply to a chunk or object, separated by commas. (For example, text that is italic, bold, and underlined has a textStyle of `italic,bold,underline`.) The following example removes the `bold` style from a chunk of text in a field:

```
get the textStyle of word 4 of field "Example"  
if "bold" is among the items of it then  
  delete item (itemOffset("bold",it)) of it  
  set the textStyle of word 4 of field 1 to it  
end if
```

How to remove a window's close box

See Also:

How to close a modal dialog box, How to close a window, How to remove a window's title bar, How to respond to closing a window, About windows, palettes, and dialog boxes, closeBox property, decorations property

You remove a stack window's close box by setting the stack's closeBox property to false, or by setting its decorations property. You can set these properties in a handler or the message box, or set the decorations in the stack's property inspector.

Modal dialog boxes normally don't have a close box. For other window types (editable windows, modeless dialog boxes, and palettes), the following statement removes the close box:

```
set the closeBox of stack "My Stack" to false
```

How to remove a window's title bar

See Also:

How to change a window's title, How to create a window the same size as the screen, How to remove a window's close box, How to show or hide a window, decorations property, windowShape property

You show a stack window without a title bar at the top by setting the stack's decorations property to empty. All windows in a Revolution application—editable window, modal dialog box, modeless dialog box, or palette—are stacks, and all of them have a decorations property you can set.

You can set the stack's decorations to empty in the message box or a handler, or in the Basic Properties pane of the stack's property inspector.

Note: Setting a stack's windowShape property to an image ID hides the stack window's title bar.

How to remove all styles from text

See Also:

How to change the style of text, How to remove a single style from text, How to set the font, style, and size of text to the default, Recipe for handling selection of choices from a Text menu, Shortcut to remove font changes from text, Text menu > Plain, plain keyword, textStyle property

The style information of text in a field is stored in the textStyle property. You remove style information—such as bold and italic—by either selecting the text and choosing Text menu>Plain or setting the chunk's textStyle property to `plain`:

set the textStyle of char 12 to 24 of field x to plain

Note: This statement sets the text style to plain, even if the field has a different style. For example, if the field's textStyle is `underline`, the statement will leave the specified text plain instead of underlined. To set the style to the default of the field (or its owners), set the textStyle to empty instead.

Tip: To change the textStyle of all the characters in a field, use the chunk expression `char 1 to -1 of field`.

How to remove subscripting and superscripting

See Also:

How to change the size of text, How to change the style of text, Text menu > Subscript, Text menu > Superscript, fixedLineHeight property, textShift property, textSize property

You set the textShift property to make text in a field subscripted or superscripted. To remove any subscripting or superscripting from text, set its textShift property to zero.

The following example makes a superscripted word in a field into a normal word (neither superscript nor subscript):

set the textShift of word 5 of field "Formulas" to zero

How to remove the box around found text

See Also:

How to search a stack, Recipe for a Find field, Edit menu > Find and Replace..., empty constant, find command, foundChunk function, foundField function, foundLine function, foundLoc function, foundText function, lock screen command, unlock screen command

When the find command locates text, it draws a box around the found text to help the user locate it on the screen. If your handler marks the found text in some other way, or if you don't want the user to see the box, you can use the find command with a value of empty to remove it:

```
lock screen -- prevents box from showing momentarily
find myString
find empty -- removes box
unlock screen
```

Note: The statement find empty also resets the found text, clearing functions such as the foundChunk. The next find search for the same string will find the first instance again.

How to remove the text from a field

See Also:

How to allow copying text from a locked field, How to clear the clipboard, How to deselect the selected text, How to prevent changing a field's text, How to respond to a change in field contents, How to select text in a field, About chunk expressions, About containers, variables, and sources of value, empty constant

You remove the text from a field by putting empty into the field:

```
put empty into field "My Field"
```

Tip: You can remove the contents of a chunk of a field by putting empty into the chunk:

```
put empty into line 3 of field 1 -- leaves a blank line
put empty into word 5 to 19 of field "My Field"
put empty into item 2 of field ID 2345
```

How to remove the underline from linked text

See Also:

How to change the style of text, Why is some text blue and underlined?, Text menu > Link, hide groups command, link keyword, linkColor property, underlineLinks property

Text whose textStyle is set to `link`, by default, is underlined. The setting of the underlineLinks property controls whether this underline is shown.

You can either set this property in a handler or the message box, or use the Basic Properties pane in the stack's property inspector to change a stack's underlineLinks.

The underlineLinks is also a global property whose setting applies to all stacks. (A stack's own underlineLinks property, if set, overrides the global setting.) To turn off link underlining for all stacks whose underlineLinks is not set, enter the following in the message box:

```
set the underlineLinks to false
```

How to rename a custom property set

See Also:

How to create a custom property set, How to delete a custom property set, How to duplicate a custom property set, How to rename a custom property, About custom properties and custom property sets, Why doesn't a custom property appear in the property inspector?, customProperties property, customPropertySet property, customPropertySets property

You rename a custom property set by selecting it and clicking Rename in the Custom Properties pane of the property inspector, or by writing a handler.

In Transcript, you can't directly rename a custom property set. Instead, you create the new custom property set with the same properties and values as the old one, then delete the old set:

```
set the customProperties["newSet"] of this card
```

```
to the customProperties["oldSet"] of this card
```

```
get the customPropertySets of this card
```

```
set the wholeMatches to true
```

```
delete line lineOffset("oldSet",it) of it
```

```
set the customPropertySets of this card to it
```

How to rename a custom property

See Also:

How to create a custom property, How to delete a custom property, About custom properties and custom property sets, Why doesn't a custom property appear in the property inspector?, customKeys property, customProperties property, wholeMatches property

You rename a custom property by selecting it and clicking Rename in the Custom Properties pane of the property inspector, or by writing a handler.

In Transcript, you can't directly rename a custom property. Instead, you create the new custom property with the same value as the old one, then delete the old custom property:

set the newProp of button "My Button" to the oldProp

of button "My Button" -- creates newProp
get the customKeys of button "My Button"
set the wholeMatches to true
delete line lineOffset("oldProp",it) of it
set the customKeys of button "My Button" to it

How to rename a file

See Also:

How to change the file type of an existing file, How to delete a file, How to determine whether a file exists, How to include a slash in a file path, How to move a stack to another file, rename command

You use the rename command to change a file's name. The following statement changes the name of a file called "Mars", and located in the current folder, to "Venus":

```
rename file "Mars" to "Venus"
```

To rename a file that's not in the current folder, include the file's path along with its old and new names.

Tip: You can also use the rename command to move a file. The following statement moves a file from the folder "Sol" to the folder "E. Eridani", without changing its name:

```
rename file "/Cosmos/Sol/Mars" to "/Cosmos/E. Eridani/Mars"
```

How to report a bug or request a feature

See Also:

How to request technical support, About what's new in version 2.0-2.0.2, Help menu > Revolution Support

You report a bug to Runtime Revolution through the bug reporting system at <http://www.runrev.com/Revolution1/bugzilla/>, or by sending email to support@runrev.com. (Please check the bug reporting system first to see whether the issue is already known.)

Your report will be more helpful if you can describe the problem in as much detail as possible. Please include:

- ò the platform and operating system version you were using
- ò the steps that caused the problem to appear
- ò whether the problem is reproducible

Tip: If you have a technical support contract, choose Help menu>Revolution Support to give your report priority handling.

How to request technical support

See Also:

How to report a bug or request a feature, Help menu > Revolution Support

You can request technical support by choosing Help menu>Revolution Support or by sending email to support@runrev.com. Be sure to include your name, organization, and license key number, and please describe the problem or question in as much detail as possible.

Runtime Revolution provides òup-and-runningö support to help you get the product working. If your question is more advanced, and you donÆt have a technical support contract, you will find helpful experts on the Revolution mailing list. You can subscribe to the list at <http://www.runrev.com/revolution/developers/maillinglists/>.

Tip: Si vous faites partie de la communautÚ Francophone, et prÚfÚrez obtenir de lÆaide ou support en Franþais, envoyez un email Ó Frederic Rinaldi <frederic@runrev.com>.

How to respond to a change in field contents

See Also:

How to respond to a drag and drop, How to respond to a keystroke, How to prevent changing a field's text, How to prevent entering certain characters in a field, How to put text into a field, Recipe for storing a modification timestamp, closeField message, exitField message, openField message

To perform an action when the user changes the text in a field, you handle the closeField message:

```
on closeField
  set the hilite of button ôChangedö to true
end closeField
```

The closeField message is sent when the user leaves a field (by clicking or tabbing out of it) and the field's contents have been changed. The closeField message is sent only when the user changes a field's text, not when a script uses the put command to change the contents of the field.

If the field's contents weren't changed, an exitField message is sent instead.

How to respond to a Control-click or right-click

See Also:

How to find out where the user clicked, How to simulate a mouse click, controlKey function, controlKeyDown message, mouseDown message, mouseUp message

To perform an action when the user Controls-clicks (Mac OS or OS X) or right-clicks (Unix or Windows), you handle the mouseDown message and check the message parameter, which specifies which mouse button was used. The following example responds to a Control-click or right-click on a field:

```
on mouseDown theButton
  if theButton is 3 then -- Control-click, right-click
    set the lockText of me to not the lockText of me
  else pass mouseDown
end mouseDown
```

Tip: You can use the same technique to find out which mouse button was used in mouseUp, mouseDoubleDown, mouseDoubleUp, and mouseRelease handlers.

How to respond to a drag and drop

See Also:

How to respond to a change in field contents, How to start a drag and drop, Recipe for dragging a background color onto an object, acceptDrags property, dragData property, dragDrop message, dragEnd message

To perform an action when the user drops data onto a locked field or another object type, you handle the dragDrop message.

The dragDrop message is sent when the user drops data on an object.

You must also set the acceptDrop property to true before a drop will be allowed. Usually, you set this property to true in a dragEnter handler.

Note: Revolution automatically handles the mechanics of dragging and dropping text between and within unlocked fields. To support this type of drag and drop operation, you don't need to do any scripting.

How to respond to a keystroke

See Also:

How to respond to a change in field contents, How to respond to mouse wheel actions, Why can't I use the arrow keys when editing a field?, Why doesn't the Tab key move to the next tab stop?, Recipe for auto-indenting in a field, Recipe for limiting the number of characters in a field, Recipe for rejecting non-digit characters typed into a field, keyDown message, keyUp message, rawKeyDown message, rawKeyUp message, type command

To perform an action when the user presses a key, you handle the keyDown message.

The keyDown message is sent when the user presses an ordinary key (one that corresponds to a character). Other messages, sent when the user presses special keys, include tabKey, returnKey, enterKey, deleteKey, backspaceKey, arrowKey, functionKey, and escapeKey. Handle these messages instead if you want to respond to the user pressing these special keys.

Note: Trapping the keyDown message and not passing it prevents typing in a field.

How to respond to a menu choice

See Also:

How to change a menu's contents when it is opened, How to determine the current selection in a menu, How to respond to changing the current setting in a combo box, How to simulate a menu choice, About menus and the menu bar, Recipe for a Cards menu, Recipe for checkmarking a menu item, Recipe for handling selection of choices from a Text menu, Tools menu > Menu Builder, menuMode property, menuPick message

To perform an action when the user chooses an item from a menu, you handle the menuPick message.

The menuPick message is sent when the user chooses a menu item from a popup menu or pulldown menu, or changes the setting in an option menu, combo box, or tabbed button. The parameter of the menuPick message is the name of the menu item the user chose.

Tip: For cascading menus, the parameter consists of the submenu's name and the menu item, separated by a space.

How to respond to a mouse click

See Also:

How to respond to a Control-click or right-click, How to respond to scroll wheel actions, How to simulate a mouse click, How to throw away unwanted mouse clicks, Why aren't mouse messages sent?, Why does a mouseUp handler fail to work?, Recipe for Hello World, click command, mouseDoubleDown message, mouseDoubleUp message, mouseDown message, mouseUp message

To perform an action when the user clicks, you handle the mouseUp message. When the user clicks an object, a mouseUp message is sent to the object.

The following handler beeps when the mouse is clicked. For example, if you place this handler in a button's script, the button beeps whenever it's clicked:

```
on mouseUp
  beep
end mouseUp
```

How to respond to changing the current setting in a combo box

See Also:

How to change a menu's contents when it is opened, How to change the selected item in an option menu or combo box, How to determine the current selection in a menu, How to respond to a menu choice, closeField message, combobox keyword, menuMode property, menuPick message, style property

When the user types a new setting into the text box of a combo box menu, Revolution sends a closeField message to the combo box. If the user chooses an item from the combo box menu, a menuPick message is sent instead. To respond to all changes in a combo box, handle both these messages.

Note: A combo box menu is a button whose menuMode is `comboBox` and whose style is `menu`. Normally, only fields receive closeField messages, but since combo box menus are functionally similar to fields, they also receive field messages.

How to respond to closing a window

See Also:

How to close a window, How to respond to moving a window, How to respond to resizing a window, close command, closeStack message, closeStackRequest message

To perform an action when a stack is closed, you handle the closeStack message.

The closeStack message is sent when the user or a handler closes a stack window. Any actions in a closeStack handler thatÆs in the current card of that stack, or an object in the current cardÆs message path, are performed at that time. When any closeStack handler finishes, the window closes.

If you want to prevent the stack from being closed—for example, if you want to give the user the option to cancel the close action—handle the closeStackRequest message instead.

Tip: To prevent these messages from being sent when using the close command in a handler, first set the lockMessages property to true.

How to respond to double-clicking a file for Mac OS or OS X

See Also:

How to assign an icon to a Mac OS standalone application, How to assign creator and type signatures to a new file, How to associate files with a Mac OS standalone application, How to change the file type of an existing file, How to do a task when starting up the application, How to respond to double-clicking a file for Windows, About file types, application signatures, and file ownership, Recipe for processing files opened in a Mac OS or OS X application, `appleEvent` message, `preOpenStack` message, `openStack` message, `startup` message

When you double-click a document file, it automatically opens in the application itÆs associated with. If you have previously associated a file type with your application, double-clicking a file with that file type automatically launches your application.

If the file is a stack file (that is, if it was created with the `save` command), your application opens it automatically. You can put any special handling the stack file requires in a `preOpenStack` handler.

If the file is any other type of file, double-clicking it or dropping it onto the applicationÆs icon sends an `appleEvent` message to the application. To display or otherwise use such a file, you handle the `appleEvent` message. For example code, see the topic “Recipe for processing files opened in a Mac OS or OS X application” in the Transcript Cookbook section of the documentation.

How to respond to double-clicking a file for Windows

See Also:

How to assign an icon to a Mac OS standalone application, How to assign creator and type signatures to a new file, How to associate files with a Mac OS standalone application, How to change the file type of an existing file, How to do a task when starting up the application, How to respond to double-clicking a file for Windows, About file types, application signatures, and file ownership, Recipe for processing files opened in a Windows application, preOpenStack message, openStack message, startup message

When you double-click a document file, it automatically opens in the application itÆs associated with. If you have previously associated a file extension with your application, double-clicking a file with that extension automatically launches your application.

If the file is a stack file (that is, if it was created with the save command), your application opens it automatically. You can put any special handling the stack file requires in a preOpenStack handler.

If the file is any other type of file, double-clicking it or dropping it onto the applicationÆs icon places the fileÆs path in a special variable with the name `ô$1ö`. To display or otherwise use such a file, you use the value of the `$1` variable in a startup handler. For example code, see the topic `ôRecipe for processing files opened in a Windows applicationö` in the Transcript Cookbook section of the documentation.

How to respond to mouse wheel actions

See Also:

How to respond to a mouse click, rawKeyDown message, rawKeyUp message

To perform an action when the user turns the mouse wheel, you handle the rawKeyDown message.

When the mouse wheel is turned downward, Revolution sends a rawKeyDown message with the parameter 65308. If the wheel is turned upward, the parameter is 65309. The following handler displays a message in the message box when the mouse wheel is used:

```
on rawKeyDown theKeyNumber
  if theKeyNumber is 65308 then -- mouse wheel down
    put "Mouse wheel down"
  else if theKeyNumber is 65309 then -- mouse wheel up
    put "Mouse wheel up"
  else pass rawKeyDown
end rawKeyDown
```

How to respond to moving a window

See Also:

How to center a window on the screen, How to change the size and position of a window, How to keep a window on top of other windows, How to prevent the user from moving a window, How to respond to closing a window, How to respond to resizing a window, Why aren't window positions saved in my application?, location property, moveStack message, rectangle property

When the user moves a window, a moveStack message is sent to the current card of the stack whose window was moved. You respond to a window being moved by writing a handler for this message:

```
on moveStack newStackH,newStackV
  -- do something
end moveStack
```

How to respond to quitting an OS X application

See Also:

How to do a task when starting up the application, How to quit the application, About menus and the menu bar, menuPick message, quit command, shutdownRequest message

On OS X systems, the `Quit` menu item is part of the Application menu, which is displayed by the operating system rather than by the application. Because of this, choosing `Quit` on OS X systems does not send a `menuPick` message, so you cannot handle quitting in a `menuPick` handler.

Instead, choosing `Quit` sends an Apple Event (class `AEvtQuit`, ID `quit`) to the application. If you don't intercept this Apple Event in an `appleEvent` handler, Revolution sends a `shutdownRequest` message in response to the Apple Event. To respond to the user choosing `Quit`, handle either of these messages.

Tip: For easiest cross-platform development, place all the code you want to execute on quitting in a `shutdownRequest` handler.

How to respond to resizing a window

See Also:

How to automatically adjust objects when a window is resized, How to change the size and position of a window, How to prevent the user from resizing a window, How to respond to closing a window, How to respond to moving a window, Geometry Management Tutorial, height property, resizeMode message, width property

To perform an action when the user changes the size of a window, you handle the resizeMode message.

The resizeMode message is sent when the user finishes resizing a stack window. Any actions in a resizeMode handler thatÆs in the current card of that stack, or an object in the current cardÆs message path, are performed at that time. When any resizeMode handler finishes, the window is displayed at the new size.

Tip: The screen is locked while a resizeMode handler is running, so it is not necessary to use the lock screen command to prevent changes from being seen.

How to respond to the user clicking a tab in a tabbed window

See Also:

How to switch between button tabs, Object menu > New Control > Tabbed Button, menuHistory property, menuMode property, menuPick message, tabbed keyword

A tabbed window is a window that contains a set of tabs (usually at the top of the window) which you click to switch between sets of controls. In Revolution, a set of tabs is implemented as a button whose style property is `menu` and whose menuMode property is `tabbed`.

When the user clicks a tab, a menuPick message is sent to the tabbed button. You respond to a click on a tab by writing a handler for this message:

```
on menuPick newTab,oldTab
    -- do something
end menuPick
```

Tip: Setting the button's menuHistory property automatically sends a menuPick message to the button.

How to respond when a card is visited

See Also:

How to change the order of cards in a stack, How to create a new card, How to do a task when starting up the application, How to go to another card, How to respond to closing a window, How to respond when a stack opens, How to scan the cards in a stack, Recipe for a Cards menu, Shortcut to go to the next card, Shortcut to go to the previous card, go command, openCard message, preOpenCard message

To perform an action when going to a card, you handle the openCard or preOpenCard message.

The preOpenCard message is sent before the card is displayed, and the openCard message is sent after it is shown. If you want to make changes to the card's appearance when visiting it, make the changes in the preOpenCard handler.

Tip: To prevent these messages from being sent when moving between cards, set the lockMessages property to true.

How to respond when a stack opens

See Also:

How to do a task when starting up the application, How to open a stack, How to respond to double-clicking a file for Mac OS or OS X, How to respond to double-clicking a file for Windows, Why does Revolution ask to purge a stack?, Why is there already a stack with the same name?, File menu > Open Stack..., go command, openCard message, openStack message, preOpenCard message, preOpenStack message

To perform an action when a stack is opened, you handle the openStack or preOpenStack message.

When a stack opens, a preOpenStack message is sent to the card that is going to be displayed, followed by an openStack message. (If you open the stack by going to a particular card in it, the messages are sent to this card. Otherwise, they are sent to the stack's first card.)

The preOpenStack message is sent just before the stack appears, so to perform an action before the stack appears, write a preOpenStack handler. The openStack message is sent just after the stack appears.

Tip: To prevent these messages from being sent when opening a stack in a handler, first set the lockMessages property to true.

How to restore the default colors of an object

See Also:

How to change an object's color, How to change the background color of text, How to change the color of text, How to find out the actual color of an object, How to make an object the same color as its owner, About colors and color references, backgroundColor property, borderColor property, bottomColor property, colors property, effective keyword, focusColor property, foregroundColor property, hiliteColor property, owner property, shadowColor property, topColor property

You restore an object's default colors by setting the object's colors property to empty.

An object inherits its colors from the object that owns it: a button inherits its colors from the card it's on, a grouped field inherits its colors from the group, a card inherits from the stack it's in, and so on. You change the colors by setting the object's eight color properties. If a color property of the object is empty—that is, if it's not set to a color—the owning object's color shows through.

Setting the colors property sets all eight of the color properties, so using it is a shortcut. You can also set the object's eight color properties (backgroundColor, foregroundColor, borderColor, topColor, bottomColor, hiliteColor, shadowColor, and focusColor) individually, either in a handler or the message box.

How to retrieve headers from an HTTP request

See Also:

How to access the Internet from behind a firewall, How to display a web page in a field, get command, http keyword, libURLLastRHHeaders function, post command, URLStatus function

When you request a web page or perform another HTTP transaction, the server sends back a header along with the information. The header is not normally displayed by browsers, but contains information about the request, such as the HTTP result code, the date the web page was last changed, and so on. (The exact content of the header depends on the server.)

You use the libURLLastRHHeaders function to retrieve the headers from the latest HTTP transaction, as in the following example:

```
get URL "http://www.example.net/index.html"  
put line 1 of libURLLastRHHeaders() into myResultCode  
if word 2 of myResultCode is 200 then displayPage(it)
```

How to return an array from a function

See Also:

How to return multiple values from a handler, About commands and functions, [] keyword, return control structure

To return an array from a function, you build the array in a variable and then return the variable, as in the following boring example:

```
function exampleArray
  put "this" into myArray[1]
  put "that" into myArray[2]
  return myArray
end exampleArray
```

In the calling handler, put the returned value into a variable to work with its individual elements:

```
put exampleArray() into myVar
put myVar[1] into field "This value"
```

How to return an error message from a handler

See Also:

About commands and functions, Recipe for converting between characters and ASCII values, return control structure

To send back an error message from a custom command, use the return control structure in the command's handler:

```
on doCustomCommand
  -- .. if there's an error:
  return "Error" -- whatever error message you want
end doCustomCommand
```

When you check the result function after executing the command, it holds the error message, if one was returned:

```
doCustomCommand
if the result is "Error" then beep
```

How to return multiple values from a handler

See Also:

How to access a value in an array variable, How to return an array from a function, About commands and functions, About containers, variables, and sources of value, @ keyword, return control structure

The return control structure, which is the normal way to pass a value back from a handler to the calling handler, only takes one value. Since the return control structure stops execution of the handler, it isn't possible to use it several times to pass several values. If you need to pass back more than one value from the same handler, try one of the following methods:

- ò Put the values you want to return into an array, and use the return control structure to pass the array back to the calling handler.

- ò Put each value you want to return in a different line of a variable, then return the variable.

- ò Pass variables by reference using the @ keyword. Variables passed by reference can be changed by the called handler.

How to save a stack

See Also:

How to store preferences or data for a standalone application, Why can't I find a stack I just saved?, Why can't I save a stack?, File menu > Save, File menu > Save As..., Shortcut to see a contextual menu, save command

You save a stack by choosing File menu>Save, or by using the save command in a handler or the message box.

The "Save" menu item saves the stack named by the topStack function. Normally, this is the frontmost stack window. If the stack has not yet been saved, "Save" acts like "Save As".

Saving a stack also saves changes to any other stacks in the same stack file.

Tip: To save palettes and dialog boxes, use Command-Shift-Control-click (Mac OS or OS X) or Control-Shift-right click (Unix or Windows) to display a contextual menu.

How to save and re-use Distribution Builder configurations

See Also:

How to create a standalone application, File menu > Distribution Builder, File menu > Open Distribution... (Distribution Builder), File menu > Save Distribution As... (Distribution Builder)

When working on an application, you may need to build test versions several times. To speed up the process of repeatedly building an application, you can save the Distribution Builder settings you use.

To save the current Distribution Builder settings, open the Distribution Builder window and set up the configuration you want. Then choose File menu>Save Distribution As and choose a name and location for your settings file.

To use the stored settings, open the Distribution Builder and choose File menu>Open Distribution. The settings from the file you select are loaded into the Distribution Builder, and will be used when you click **Build Distribution**.

How to save changes to a SQL database

See Also:

How to close the connection to a database, How to make changes to a SQL database, About connecting to and using SQL databases, `revCommitDatabase` command, `revExecuteSQL` command, `revRollBackDatabase` command

You use the `revCommitDatabase` command to save changes youÆve made to a database.

The `revCommitDatabase` command issues a SQL COMMIT statement to the database. The COMMIT statement saves any changes youÆve made to the database since the last COMMIT.

Tip: To undo changes to a database, use the `revRollBackDatabase` command.

How to save changes to records from an automatic database query

See Also:

How to display a single record from an automatic database query, How to move through the records in a record set (database cursor), How to save changes to a SQL database, How to set up an automatic database query, How to specify a primary key for an automatic database query, About connecting to and using SQL databases, `revMoveToFirstRecord` command, `revMoveToLastRecord` command, `revMoveToNextRecord` command, `revMoveToPreviousRecord` command

You use the Database pane in a button's Properties palette to save records associated with an automatic database query. Follow these steps:

1. Create or select a button and choose Object menu>Object Inspector, then choose "Database" from the menu at the top of the inspector.
2. From the Query menu, choose the automatic query you want to use.
3. From the Action menu, choose "Update Record". When you click the button, the current record is saved to the database.

Important! You must have selected a primary key on the Record Set tab of the Database Query Builder. If there is no primary key, or the primary key is not unique for each record, the data in the database may be corrupted when you update it.

How to scan the cards in a stack

See Also:

How to list the cards in a stack, Tools menu > Application Browser, show cards command

You use the show cards command to scan the cards in a stack, one after the other. You can either scan a specified number of cards starting from the current card, or flip through all the cards in the current stack.

You use the Application Browser to see a list of all the cards in a stack. To see the list of cards, choose Tools menu>Application Browser and find your stack in the list. Click the disclosure button next to the stack's name to see a list of all cards in that stack.

How to schedule a future message

See Also:

How to cancel a pending message, How to determine which pending messages have been scheduled, Recipe for a card slideshow, send command, time function

You schedule an event by using the send command to send a message after a specified delay. For example, to make the computer beep after 60 seconds, place this statement in a handler:

```
send "beep" to this card in 60 seconds
```

If you want to do a more complex task after a delay, write a message handler that performs the task, then send the handler's name to the object whose script contains the handler:

```
send "myCustomMessage" to button "Container" in 5 ticks
```

How to scroll a field

See Also:

How to create a scrolling window, Recipe for a "roll credits" effect, scrollbarDrag message, style property, vScroll property

You scroll a field by setting its vScroll property. (To scroll horizontally, use the hScroll property.) The following example scrolls a field downward 20 pixels:

set the scroll of field 1 to (the scroll of field 1 + 20)

When the user clicks in the scroll bar to scroll a field, the setting of the field's vScroll property changes automatically.

Tip: A field's maximum scroll is equal to its formattedHeight minus its height. To scroll a field to the bottom, use a statement like the following:

set the scroll of field 1 to

the formattedHeight of field 1 - the height of field 1

How to search a container

See Also:

How to search a stack, Edit menu > Find and Replace..., itemOffset function, lineOffset function, offset function, wordOffset function

You find a string's location in a container (such as a variable, property, field, or URL) by using the offset function. The offset function returns the location of the string's first character. If the string is not found in the container, the offset function returns zero. This example checks whether the word "Gotcha" appears in a URL:

```
if offset("Gotcha", URL "http://www.example.com/")  
  
    is zero then answer "Not found!"
```

How to search a stack

See Also:

How to remove the box around found text, How to search a container, Recipe for a Find field, Edit menu > Find and Replace..., dontSearch property, find command, foundChunk function, foundText function

You search a stack by choosing Edit menu>Find and Replace, or by using the find command in a handler or the message box. In the Find and Replace dialog box, you can search the properties, scripts, or text of all the objects in a card, stack, or stack file.

Tip: To simply locate the next occurrence of text in a field, enter the following command into the message box:

```
find "text to find"
```

The find command displays the next card containing the text. (To find a phrase, use find whole instead of find.)

How to search global variables

See Also:

How to declare a variable, How to delete a variable, How to search a container, How to search a stack, About containers, variables, and sources of value, Edit menu > Find and Replace..., Development menu > Variable Watcher, globalNames function

You can search variables in a number of ways.

To search the contents of all global variables, choose Edit menu>Find and Replace, then choose ôGlobal Variable Contentsö from the In menu near the top of the Find and Replace window.

To search for a global variable by name, use the globalNames function:

```
if "myGlobal" is among the items of the globalNames...
```

Tip: To see a list of global variables, choose Development menu>Variable Watcher.

How to see and change an object's properties

See Also:

How to show more than one object's properties at a time, Getting Started Tutorial, Object menu > Object Inspector, Object menu > Card Inspector, Object menu > Stack Inspector, Shortcut to display a contextual menu, Shortcut to display a contextual menu (Pointer tool), properties property

You see and change an object's properties in the object's property inspector. To open the property inspector, select the object with the Pointer tool, then choose Object menu>Object Inspector. (For a stack or card, choose Object menu>Stack Inspector or Object menu>Card Inspector.)

The property inspector has a menu at the top. You can select any of several panes from this menu to see and change various groups of properties.

You can also set any of an object's properties in a handler (or in the message box) using the set command.

How to select a card

See Also:

How to select a card, Edit menu > Cut, Edit menu > Copy, Edit menu > Clear, Edit menu > Duplicate, select command, selectedObject function

You select a card by double-clicking it with the Pointer tool, or by using the select command in a handler or the message box.

Tip: Make sure to click a part of the card thatÆs not covered by a control: double-clicking a control instead selects it and displays the controlÆs property inspector.

Once you have selected a card, you can use the Edit menu to cut, copy, or duplicate the card.

How to select a control that's part of a group

See Also:

How to select a group, Edit menu > Select Grouped Controls, Object menu > Edit Group, Shortcut to reverse Select Grouped Controls, `selectGroupedControls` property

When you click a grouped control, the entire group is selected so that you can change the group's properties, and move or resize it.

To select the control itself, you set the `selectGroupedControls` property to true. You can either set this property in a handler or the message box, or choose Edit menu>Select Grouped Controls so that the menu item is checked.

Tip: To temporarily reverse this setting, hold down the Option key (on Mac OS or OS X systems) or the Alt key (on Unix or Windows systems) while clicking a control.

How to select a group

See Also:

How to select a control thatÆs part of a group, Edit menu > Select Grouped Controls, Shortcut to reverse Select Grouped Controls, select command, selectGroupedControls property

You select a group by setting the selectGroupedControls property to false and then clicking any object the group, or by using the select command in a handler or the message box.

To set the selectGroupedControls to false, make sure the ôSelect Grouped Controlsö menu item in the Edit menu is not checked. You can also set this property in a handler or the message box.

The following statement selects a group:

```
select group "My Group"
```

Tip: Hold down the Command key (on Mac OS or OS X systems) or Control key (on Unix or Windows systems) while clicking, to momentarily reverse the ôSelect Grouped Controlsö setting.

How to select an entry in a list field

See Also:

How to determine which line of a list field is selected, Why can't I leave all lines of a list field unselected?, hilitedLine property, listBehavior property, multipleHilites property, noncontiguousHilites property, toggleHilites property

A list field is a field whose listBehavior property is set to true, making the field a clickable list. Clicking a line in such a field selects the line.

You select a line in a list field by setting the field's hilitedLines property in a handler or the message box. To select an entry, set the hilitedLines to the line number you want to select. For example, to select the fourth line in a list field called "My List", use the following statement:

```
set the hilitedLine of field "My List" to 4
```

If the field's multipleHilites property is true, you can select multiple lines. To select multiple entries in a list field, separate the line numbers with commas:

```
set the hilitedLines of field "My List" to "2,4,5"
```

How to select text in a field

See Also:

How to allow copying text from a locked field, How to prevent changing a field's text, How to put text into a field, Why can't I select, copy, or paste text?, Why is the selection lost when clicking a button?, lockText property, select command, traversalOn property

You use the select command to select text in a field.

Tip: If the field's traversalOn property is true, the user can select text and copy it, but not change it. If the traversalOn is false, the user cannot select text in the field.

The traversalOn does not affect use of the select command in a handler or the message box—only the user's actions.

To select a line or lines in a list field, you set the list field's hilitedLine property.

How to send mouse messages to an unlocked field

See Also:

How to allow copying text from a locked field, How to deselect all the entries in a list field, How to find out what text the user clicked, How to respond to a Control-click or right-click, How to select text in a field, How to simulate a mouse click, Why aren't mouse messages sent?, Recipe for toggling a field between locked and unlocked, lockText property, mouseDown message, mouseUp message

Usually, when you click a field whose lockText property is false, messages such as mouseDown and mouseUp aren't sent. This is to allow users to edit the field's contents without extraneous handlers running.

If you want to send the usual mouse message to an unlocked field, Control-click (Mac OS or OS X) or right-click (Unix or Windows) in the field.

How to set fonts for all the objects in a card or stack

See Also:

How to change the font of text, About properties and property profiles, effective keyword, owner property, textFont property

If a field's textFont property is empty, the field inherits the font of the card it's on. If no font is specified for the card, the stack's font is used, and so on. If you set a the textFont of a stack, every object in it will automatically use that font, without your having to set it for each object. For example, to use the font "Comic Sans" for all the text in a stack, set the stack's textFont to "Comic Sans".

Important! Inheritance is used to determine an object's font only if the object itself has no font setting. If an object's textFont is non-empty, that setting overrides any setting the object might inherit from an object above it in the object hierarchy.

Tip: The textSize, textStyle, and foregroundColor are also inherited properties, so you can set the size, style, and color of text throughout a stack by setting these properties for the stack.

How to set text in a field to the field's default font

See Also:

How to change the font of text, How to remove all styles from text, How to restore the default colors of an object, How to set fonts for all the objects in a card or stack, How to set the font, size, and style of text to the default, Text menu > Font > Use Owner's Font, effective keyword, textFont property

By default, text in a field is displayed in the font specified by the field's textFont property, but chunks of a field can be changed to another font. To set the font of a chunk back to the field's textFont, you set the chunk's textFont to empty. (This also sets the chunk's style and size to the field's default.)

The following statement changes a part of the text in a field to use the field's default font:

set the textFont of word 5 of field "My Field" to empty

Tip: Choosing Text menu>Font>Use Owner's Font is equivalent to setting the text's textFont property to empty.

How to set the font, style, and size of text to the default

See Also:

How to change the style of text, How to remove a single style from text, How to remove all styles from text, Recipe for handling selection of choices from a Text menu, Shortcut to remove font changes from text, Text menu > Font > Use Owner's Font, effective keyword, textFont property, textSize property, textStyle property

To remove the font, size, and style information from a chunk or object, you set the textFont property to empty. (This removes the style and size along with the font.) You can either set this property in a handler or the message box, or choose Text menu>Font>Use Owner's Font.

Removing the font, size, and style information from a chunk of text in a field causes the text to be shown with the field's default font, size, and style. Similarly, removing this information from an object causes its text to be shown with the font, size, and style of the object's owner.

The following statement sets some of the text in a field to the field's default appearance:

```
set the textFont of word 1 to 4 of field 1 to empty
```

How to set up an automatic database query

See Also:

How to display a single record from an automatic database query, How to display records from an automatic database query as a table, How to save changes to records from an automatic database query, About connecting to and using SQL databases, Database Types Reference, Tools menu > Database Query Builder, revOpenDatabase function, revQueryDatabase function, revQueryDatabaseBLOB function

You use the Database Query Builder to set up an automatic database query, which is stored in a stack. To create an automatic query, follow these steps:

1. Choose Tools menu>Database Query Builder.
2. Click the New (+) button at the top to create an automatic query, then enter the settings for the new query and click Connect.
3. Click the Record Set tab and choose a table from the menu. A SQL query that gets all the records in the table is entered automatically. (You can change it to any valid SQL query.)

When you close the Database Query Builder, the automatic query is saved in the stack.

How to show more than one object's properties at a time

See Also:

Object menu > Object Inspector, Object menu > Card Inspector, Object menu > Stack Inspector,
Shortcut to change a control's properties

Normally the property inspector shows the properties of whatever object is selected. To open a second property inspector for another object, you must first lock the property inspector to the current object.

To lock the property inspector, click the small padlock icon at the top right of the palette.

How to show or hide a window

See Also:

How to close a window, How to hide other applications' windows, How to hide the Toolbar, How to open a stack, How to remove a window's title bar, How to show or hide the menubar, How to take over the screen, hide command, show command, visible property

You show or hide a stack window by setting the stack's visible property to true or false.

All windows in a Revolution application—editable window, modal dialog box, modeless dialog box, or palette—are stacks, and any of them can be shown or hidden using the visible property.

You can either set the property directly in a handler or the message box, or use the show and hide commands.

How to show or hide the menu bar

See Also:

How to enable or disable a menu item, How to hide the Toolbar, How to show or hide a window, How to switch between menu bars, How to take over the screen, Why do menus disappear when I switch cards?, Why don't the menus appear when I open a stack?, Development menu > Suspend Development Tools, decorations property, editMenus property, hide menubar command, show menubar command

You show and hide the menu bar differently, depending on whether the menus are inside the stack window (as is standard for Unix and Windows applications) or in a separate menu bar at the top of the screen (the standard for Mac OS and OS X systems).

To hide a menu bar inside a stack window, set the stack's decorations property to a value that does not include "menu". You can either set this property in a handler or the message box, or use the Basic Properties pane in the stack's property inspector to change a stack's decorations.

To hide a separate menu bar, use the hide menubar command in a handler or the message box. To make it visible again, use the show menubar command.

How to show or hide the task bar

See Also:

How to hide the Toolbar, How to show or hide a window, How to show or hide the menu bar, How to take over the screen, hide taskbar command, show taskbar command

You use the show taskbar and hide taskbar commands to show and hide the Windows task bar.

This capability is particularly useful for applications such as standalone kiosks, where operating-system elements such as the task bar may distract the user (and where hiding them may make the application more secure from accidental or deliberate tampering).

How to shuffle the cards in a stack

See Also:

How to change the order of cards in a stack, How to sort the contents of a field, How to shuffle the items or lines in a container, any keyword, random function, sort command

You use the sort command to shuffle the cards in a stack into a new order. The following statement shuffles the cards in the current stack randomly:

```
sort this stack by random(the number of cards)
```

Tip: Use the following statement to visit a random card in the current stack:

```
show any card
```


How to shuffle the items or lines in a container

See Also:

How to change every chunk in a container, How to shuffle the cards in a stack, How to sort a portion of a container, How to sort the contents of a field, About chunk expressions, Recipe for an Elizabethan insult generator, any keyword, random function, sort container command

You use the sort container command to shuffle the items or lines in a container into a new order. To get a random order, you use the random function as the sort key.

The following statement shuffles the items of a variable:

```
sort items of myVar numeric  
by random(the number of items in myVar)
```

The random function returns an integer between 1 and the total number of items, and one random number is assigned to each item, so the items are sorted in this random order.

How to simulate a menu choice

See Also:

How to determine the current selection in a menu, How to simulate a mouse click, menuPick message

You simulate a menu choice from within a handler by sending a menuPick message to the button that contains the menu.

Menus in Revolution—even menus in the menu bar—are contained in button objects. When you choose an item from a menu, Revolution sends the menu button a menuPick message with the name of the menu item as a parameter. Sending a menuPick message to the button is equivalent to choosing the menu item manually. This example chooses Edit menu>Preferences:

```
send "menuPick Preferences" to button "Edit"
```

```
of stack "revMenubar"
```

How to simulate a mouse click

See Also:

How to find out where the mouse pointer is, How to find out where the user clicked, How to find out which object the user clicked, How to simulate a menu choice, How to throw away unwanted mouse clicks, click command, mouseDown message, mouseUp message

You use the click command in a handler or the message box to simulate a mouse click. This example simulates a click on a button:

click at the location of button "OK"

You can specify the exact location of the click, as well as which mouse button you want to simulate and any modifier keys. This example simulates a Shift-click with mouse button 3 (the right mouse button), at the point 100 pixels from the left edge of the current stack, and 200 pixels from the top:

click button 3 at 100,200 with shiftKey

Tip: You can also trigger an object's mouseUp handler by sending the mouseUp message to the object:

send "mouseUp" to button "OK"

Sending a mouseUp message does not highlight the button, unlike clicking.

How to simulate the action of a button

See Also:

How to assign a keyboard equivalent to a button, How to find out which object the user clicked, How to make the Return or Enter key activate a button, How to simulate a mouse click, About messages and the message path, click command, mouseDoubleUp message, mouseDown message, mouseUp message

Usually, the action of a button is contained in a mouseUp handler. When you send a mouseUp message to the button, this handler is triggered and the action is performed:

```
send "mouseUp" to button "My Button"
```

To fully simulate a button click—including highlighting the button and triggering any mouseDown handler in its script, as well as mouseUp—you use the click command instead:

```
click at the loc of button "My Button"
```

Important! The click command may produce unexpected results if another object overlaps the center of the button, because the simulated click will affect the other object instead of the button.

How to sort a portion of a container

See Also:

How to shuffle the items or lines in a container, How to sort the contents of a field, About chunk expressions, sort container command

To sort a part of a container, you put that part in a variable, use the sort container command to sort the variable, then put the sorted portion back into the container.

The following example sorts lines 21-30 in a field:

```
get line 21 to 30 of field "Movies"  
sort lines of it  
put it into line 21 to 30 of field "Movies"
```

How to sort the contents of a field

See Also:

How to change the order of cards in a stack, How to shuffle the items or lines in a container, How to sort a portion of a container, Recipe for sorting a list of titles, filter command, sort command, sort container command

You use the sort container command to sort the contents of a field.

```
sort lines of field 1
sort items of field "My Field"
```

You can sort a field's lines or items by any part of each line or item using the each keyword:

```
sort lines of field ID 2349 by item 2 of each
sort lines of field myField by last word of each
```

How to speak several phrases in succession

See Also:

How to make the computer speak out loud, `revIsSpeaking` function, `revSpeak` command, `revStopSpeech` command

To speak several phrases one after the other, use the `revSpeak` command to speak each phrase, then wait until the `revIsSpeaking` function returns false before speaking the next, as in the following example:

```
repeat for each line thisSpeech in mySpeeches
  revSpeak thisSpeech
  wait while revIsSpeaking()
end repeat
answer "Speeches are finished! Time for rubber chicken!"
```

How to specify a character thatÆs not on the keyboard

See Also:

Recipe for converting between characters and ASCII values, `charToNum` function, `numToChar` function, `rawKeyDown` message, `type` command

You use the `numToChar` function to specify a character that canÆt be typed (such as a control character).

To use the `numToChar` function, you must know the characterÆs ASCII value. For example, the ASCII value of the letter `A` is 65, so the expression `numToChar(65)` evaluates to `A`.

Tip: To use the `numToChar` function to specify double-byte characters, first set the `useUnicode` property to `true`.

How to specify a primary key for an automatic database query

See Also:

How to save changes to records from an automatic database query, About connecting to and using SQL databases, Tools menu > Database Query Builder

A primary key is a database field, or combination of fields, that is different for each record in the database. The Database Query Builder uses the primary key when updating a record.

To choose a primary key, open the Database Query Builder and select the automatic query's name from the menu at the top. Then click the Record Set tab and choose the name of a database field from the Primary Key menu at the bottom of the window.

To use a combination of fields as the primary key, enter the field names in the Primary Key box, separated by commas.

Important! The primary key must be unique to each record. If you choose a primary key that is the same for two or more different records, the database may become corrupted when you update it.

How to specify what cards to include in a printed report

See Also:

How to create and store report printing settings for later use, How to create a report layout card, How to preview a printed report, How to print a report, Tools menu > Report Builder

You use the Report Builder to specify the cards you want to include in a printed report. First, choose Tools menu>Report Builder and click the Print Range tab. Then choose a selection option from the Print menu in the Report Builder window.

To select cards to print at run time, choose "Marked Cards" or "Unmarked Cards" and use the mark and unmark commands in a handler to select the cards you want to print.

To print specific cards by number, choose "A Specific Range of Cards" and enter numbers, ranges, or a combination, separated by commas.

To select cards programmatically, choose "Cards Fitting an Expression".

How to speed up display of information from the web

See Also:

How to access the Internet from behind a firewall, How to fetch data from the Internet, How to open a URL in a browser, About using URLs, uploading, and downloading, cachedURLs function, load command, unload command

To pre-fetch a document from the web for faster display, you use the load command.

The load command downloads the specified document and places it in a cache. Once a document has been cached, it can be accessed nearly instantaneously when you use its URL, because Revolution uses the cached copy in memory instead of downloading the URL again.

For best performance, use the load command at a time when response speed isn't critical (such as when your application is starting up), and only use it for documents that must be displayed quickly, such as images from the web that will be shown when you go to the next card.

Tip: To remove a document from the cache, so that the next time you access it Revolution will download it again, use the unload command.

How to start a drag and drop

See Also:

How to respond to a drag and drop, Recipe for dragging a background color onto an object, dragData property, dragStart message, mouseDown message

To begin a drag and drop operation, the user clicks and holds the mouse pointer. This sends a mouseDown message to the object.

To allow drags from a locked field or from another object type, in the object's mouseDown handler, set the dragData property to the data you want to drag. When the mouse is clicked and then moved, with a value in the dragData, a drag and drop is initiated.

Note: Revolution automatically handles the mechanics of dragging and dropping text between and within unlocked fields. To support this type of drag and drop operation, you don't need to do any scripting.

How to stop a running handler

See Also:

How to cancel a pending event, How to prevent interrupting a handler, Why can't I interrupt a handler?, allowInterrupts property, canAbort property, interrupt function

To stop a running handler, press Control-period or Control-break (on Windows or Unix) or Command-period (on Mac OS or OS X).

To stop a handler from within the handler (for example, to stop if the mouse button is no longer being pressed), use the exit or exit to top control structure:

```
if the mouse is not down then exit mouseDown
```

How to store a file in a custom property

See Also:

How to copy a resource fork, How to create a custom property, How to create a file, How to delete a custom property, How to delete a file, How to export data to a file, How to import data from a file, How to store styled text in a variable or property, About custom properties and custom property sets, About using URLs, uploading, and downloading, Why was a downloaded file corrupted?, binfile keyword, compress function, file keyword, uncompress function, URL keyword

You can use a URL to store a file's content in a custom property:

```
set the myStoredFile of stack "My Stack"  
to URL "binfile:mypicture.jpg"
```

You restore the file by putting the custom property's value into a URL:

```
put the myStoredFile of stack "My Stack"  
into URL "binfile:mypicture.jpg"
```

Tip: To save space, compress the file before storing it:

```
set the myStoredFile of stack "My Stack"  
to compress(URL "binfile:mypicture.jpg")
```

How to store an array in a custom property

See Also:

How to create an array variable, How to create a custom property, How to create a custom property set, About custom properties and custom property sets, customProperties function, customPropertySets function

It is not possible to store an array in a single custom property, because each custom property set is already an array of the custom properties in the set. However, if you store a set of custom properties in a custom property set, the set can be used just like an array. You can refer to the custom property set as though it were a single custom property, and the properties in the set as the individual elements of the array.

To store an array variable as a custom property set, use a statement like the following:

```
set the customProperties["myProperty"] of me to theArray
```

To retrieve a single element of the array, use a statement like this:

```
get the myProperty["myElement"] of field "Example"
```

How to store preferences or data for a standalone application

See Also:

How to associate files with a Mac OS standalone application, How to associate files with an OS X standalone application, How to associate files with a Windows standalone application, How to create a file, How to export data to a file, How to import and export styled text, How to import data from a file, How to save a stack, About filename specifications and file paths, About file types, application signatures, and file ownership, binfile keyword, export command, file keyword, filename property, go command, import command, read from file command, save command, specialFolderPath function, URL keyword, write to file command

A standalone application cannot save changes to itself, so you store preferences or data for a standalone application by saving the data in a separate file. There are two ways to save data:

- ò Use a stack to store and display the data, and use the save command to save the stack in a file. This approach is best if your application is based on working with documents: the saved stack file becomes your document. You re-open the stack with the go command.

- ò Put the data in a file using the file's URL as a container (or use commands such as write to file and export to save the data), then read the data from these files when your application starts up. This approach is best if the data you want to save consists of settings or similar data.

Tip: To find the Preferences folder, use the specialFolderPath function.

How to store styled text in a variable or property

See Also:

How to copy styled text between fields, How to import and export styled text, htmlText property

Fields are the only kind of container that supports styled text directly. You can put the contents of a field into a variable or property, but only the plain text is stored, not the styles.

You store styled text from a field by using the htmlText property to get the text along with the styles, as in the following statement:

```
put the htmlText of field "Storage" into myVariable
```

To display the styled text in a field, set the field's htmlText property:

```
set the htmlText of field "Result" to myVariable
```

How to switch between button tabs

See Also:

How to respond to the user clicking a tab in a tabbed window, menuHistory property, menuMode property, menuPick message, tabbed keyword

A tabbed window is a window that contains a set of tabs (usually at the top of the window) which you click to switch between sets of controls. In Revolution, a set of tabs is implemented as a button whose style property is `ômenuö` and whose menuMode property is `ôtabbedö`.

You switch between tabs by setting the button's menuHistory property to the number of the tab you want to switch to. For example, to switch to the second tab, use the following statement:

set the menuHistory of button "My Tabs" to 2

Tip: Setting the menuHistory property sends a menuPick message to the button. This is the same message that's sent when the user clicks a tab.

How to switch between menu bars

See Also:

How to enable or disable a menu item, How to show or hide the menu bar, About menus and the menu bar, defaultMenubar property, editMenus property, hide command, lockMenus property, menubar property, show command

If your application has two or more modes of operation that require different sets of menu commands, you may want to create a menu bar for each mode. You switch between menu bars differently, depending on whether the menus are inside the stack window (standard for Unix and Windows applications) or in a separate menu bar at the top of the screen (standard for Mac OS and OS X systems).

To switch the menu bar inside a stack window, show the new menu bar:

```
hide group "Editing Menus" -- hide old menu bar
show group "Viewing Menus" -- show new menu bar
```

To switch the menu bar at the top of the screen, change the stack's menubar property to the name of the menu bar group you want to use. If the stack does not have its own menu bar, change the global defaultMenubar property instead.

How to take a picture with a video camera

See Also:

How to capture video from a video camera, imageData property, revVideoFrameImage function

You take a still picture with a video camera by using the revVideoFrameImage function. while the video grabber window is previewing or recording video from the camera.

First, connect the camera, open the video grabber with the revInitializeVideoGrabber command, and display video in the video grabber with either the revPreviewVideo or revRecordVideo command. When the image in the video grabber is right, use the revVideoFrameImage to get the picture data in the video grabber.

Then set the imageData property of an image to the returned value. The image now contains a still picture from the moment you called the revVideoFrameImage function.

How to take over the screen

See Also:

How to hide other applicationsÆ windows, How to show or hide a window, How to show or hide the menu bar, View menu > Backdrop, backdrop property, hide command, hide menubar command, hide taskbar command, show command, show menubar command, show taskbar command

To show only your applicationÆs windows, and hide other applications, you use the backdrop property. This is a useful capability when designing an information kiosk or other application that is intended to be used on its own, without distraction from other applications.

To set a backdrop, you set the backdrop property to the color or pattern you want to use. The backdrop covers the entire screen behind your applicationÆs windows:

set the backdrop to "aliceblue" -- a color
set the backdrop to the ID of image "Plaid" -- a pattern

Tip: To hide the menu bar (on Mac OS and OS X systems) or the task bar (on Windows systems), use the hide menubar and hide taskbar commands.

How to temporarily remove a portion of a script

See Also:

-- keyword, /**/ keyword, How to edit an object's script, Script menu > Comment (Script Editor), Script menu > Uncomment (Script Editor), View menu > Default Comment Character (Script Editor)

When writing a script, you may want to temporarily prevent part of it from being executed. For example, you might be debugging a complex handler and decide to remove most of it in order to simplify the situation, then add the rest of the code back, line by line, as each line is debugged.

You temporarily remove parts of a script by commenting them out. To comment out one or more lines in a script, select the lines, then choose Script menu>Comment. This places a comment character (-- or #) in front of each selected line. When the code is executed, Revolution ignores all commented lines.

Tip: You can comment out an entire handler by commenting out its first line.

How to throw away unwanted mouse clicks

See Also:

How to cancel a pending message, How to find out which object the user clicked, Why does an unwanted handler run?, cancel command, flushEvents function, mouseDown message, mouseUp message

If the user continues clicking the mouse (possibly in frustration) while a lengthy handler is already running, additional mouseDown and mouseUp messages are sent when the handler finishes. To cancel user actions (such as clicks) and prevent them from triggering unwanted handlers, you use the flushEvents function. For example, to throw away pending mouseUp messages, use the following statement at the end of the handler:

```
get flushEvents(mouseUp)
```

If any events have occurred since the current handler started executing, normally they are processed when the handler ends. The flushEvents function throws away pending system events, so it's especially useful during long handlers where you don't want the user's actions to trigger the handler again.

How to trap a message before a handler is executed

See Also:

How to enter the debugger, How to intercept a message, About messages and the message path, exit control structure, pass control structure

When a message is not handled by the object itÆs sent to, itÆs passed to the next object in the message path. Occasionally, itÆs desirable to prevent a message from being passed by certain objects. You do this by placing an empty handler for the message in the objectÆs script.

For example, a card might include a mouseDown handler that should be triggered by clicking most of the controls, but not a particular button. In this case, you can trap the message with the following handler:

```
on mouseDown -- in the particular buttonÆs script
    exit mouseDown
end mouseDown
```

This handler traps the mouseDown message, preventing it from being passed to the card when this one button is clicked.

How to trigger a handler

See Also:

How to simulate the action of a button, How to use a handler outside the message path, About messages and the message path, Why does a mouseUp handler fail to work?, Why does an unwanted handler run?, send command

You trigger a handler by sending the handler's corresponding message to the object whose script the handler is in.

For example, if a stack's script contains a handler for the message `myMessage`, the following statement triggers that handler:

```
send "myMessage" to stack "My Stack"
```

If the handler's object is in the message path or is in the same object's script, you can simply use the handler's name as a command:

```
myMessage -- sends the message along the message path
```

You can use the same technique to trigger handlers for built-in messages such as `mouseUp` or `enterKey`.

How to unwedge upload and download operations

See Also:

How to access the Internet from behind a firewall, How to cancel a file transfer in progress, About using URLs, uploading, and downloading, Why can't I upload a file?, resetAll command

If you are experiencing unexplained problems uploading or downloading data, or using ftp and http URLs, resetting all Internet library operations may clear whatever is causing the problem.

You cancel all uploads and downloads in progress, close all open sockets, and clear the URL cache using the resetAll command.

Caution! The resetAll command closes all open sockets, which includes any other sockets opened by your application and any sockets in use for other uploads and downloads. Because of this, you should avoid routine use of the resetAll command. Consider using it only during development, to clear up connection problems during debugging.

How to upload a file to an FTP server

See Also:

How to cancel a file transfer in progress, How to convert line endings when transferring a file, How to download a file from an FTP server, How to remove a file from an FTP server, How to unwedge upload and download operations, How to use a password for an FTP server, About using URLs, uploading, and downloading, ftp keyword, libURLftpUpload command

You use the libURLftpUploadFile command to upload a file:

```
libURLftpUploadFile "testfile",  
  
"ftp://user:pass@ftp.example.org/test"
```

To upload data that is not stored in a file (such as the content of a field or variable), you use the libURLftpUpload command instead.

You can also put data into an ftp URL to upload it:

```
put URL "binfile:testfile"  
  
into URL "ftp://user:pass@ftp.example.org/test"
```

Note: Unlike the libURLftpUpload command, the URL method causes the handler to pause until the upload is completed.

How to use a custom WDEF resource to display a window

See Also:

On Mac OS and OS X systems, you can display a window using a custom WDEF resource by setting the `stackÆs decorations` property to the WDEF identifier. A WDEF identifier is the resource number of a WDEF resource multiplied by 16, plus a variation code between zero and 15:

```
set the decorations of me to 2039 -- WDEF 127, variation 7
```

Several WDEF resources are included in Mac OS and OS X. The exact WDEFs available vary from version to version. WDEFs include variations on title bar placement, outlines, and other aspects of window appearance. Custom WDEFs, like externals, can be compiled in a language like C.

Tip: For a cross-platform method of changing a windowÆs shape, use the `windowShape` property.

How to use a handler outside the message path

See Also:

How to call a custom function thatÆs not in the message path, About messages and the message path, call command, send command, value function

To use a message handler thatÆs not in any object in the message path, you use the send command. For example, if youÆre writing a handler for the script of card 1 and want to use a message handler thatÆs in the script of card 2, use a statement like the following:

```
send "myMessage" to card 2 -- myMessage handler in card 2
```

Tip: If an object contains handlers that you want to use repeatedly, it may be easier to include that object in the message path. To include an object in the message path of every other object, you use the start using or insert script commands.

How to use a password for a web page

See Also:

How to display a web page in a field, How to open a URL in a browser, How to retrieve headers from an HTTP request, How to use a password for an FTP server, About using URLs, uploading, and downloading, http keyword

You provide a user name and password for a web server that uses HTTP Basic Authentication by including them in the URL when you request a file or upload a file.

For example, suppose you want to get a file named `secret.html` on a server named `www.example.com`. Your user name on this server is `jane`, and your password is `ferret`. You separate the user name and password with a semicolon, and use an `@` sign to separate them from the rest of the URL:

```
get "http://jane:ferret@www.example.com/secret.html"
```

How to use a password for an FTP server

See Also:

How to create a directory on an FTP server, How to download a file from an FTP server, How to list the files in an FTP directory, How to remove a file from an FTP server, How to upload a file to an FTP server, How to use a password for a web page, About using URLs, uploading, and downloading, ftp keyword, libURLftpCommand function

You provide a user name and password for an FTP server by including them in the URL when you request a file or upload a file.

For example, suppose you want to use the libURLftpUpload command to upload "Test File" to a server named ftp.example.org. Your user name on this server is jim, and your password is aardvark. You separate the user name and password with a semicolon, and use an @ sign to separate them from the rest of the URL:

```
libURLftpUploadFile "/Disk/Folder/Test File",
```

```
"ftp://jim;aardvark@ftp.example.org/test.txt"
```

Note: If you don't specify a user name and password, Revolution adds the anonymous user name and a dummy password automatically, in accordance with the convention for public FTP servers.

How to use a SQL query to select records in a database

See Also:

How to execute a SQL statement on a database, How to find out which record sets are open in a database, How to get the contents of records in a SQL database, How to move through the records in a record set (database cursor), About connecting to and using SQL databases, revQueryDatabase function, revQueryDatabaseBLOB function

You use the revQueryDatabase function to select records in a SQL database. In the revQueryDatabase function, you provide a SQL SELECT statement that indicates which records to select. The selected records become part of a record set (database cursor), which you can refer to using the ID number returned by the function.

In the following example, the query selects all the records in a table called `Emps`, and the ID number is placed in a variable called `allEmps`:

```
put revQueryDatabase(2,"SELECT * FROM Emps") into allEmps
```

Note: If the records may contain large amounts of binary data, use the revQueryDatabaseBLOB function instead.

How to use a stack thatÆs on a web server

See Also:

How to access the Internet from behind a firewall, How to open a stack, How to open a URL in a browser, How to use a password for a web page, Recipe for going to a stack on the web, go command, http keyword, URL keyword

To open a stack thatÆs stored on a web server, you use the go command with the stackÆs URL. For example, suppose a stack file named `teststack` has been uploaded to a web server. To open this stack in your application, use a statement like the following:

```
go stack URL "http://www.example.com/teststack"
```

Revolution downloads the stack and opens it, just like a stack thatÆs stored in a file on your system.

Tip: If the stack has been compressed with gzip, you can use the decompress function to open it:

```
go stack
```

```
decompress(URL "http://www.example.com/teststack.gz")
```

How to use a stack without displaying it

See Also:

To use or modify a stack without displaying it to the user in a window, use one of the following approaches:

- ò To use or set properties in the stacks in a file, first load it into memory by getting any property of the main stack, referring to it by its filename. Once a stack file is loaded into memory, all its objects can be accessed.
- ò To visit to a stack briefly in a handler, set the lockScreen property to true before visiting the stack. When the handler is finished with the stack, close the stack window, then set the lockScreen to false.
- ò To open a stack window without displaying it, set the stackÆs visible property to false before opening it.

How to use environment variables

See Also:

How to declare a variable, How to get the location of the user's home directory, About containers, variables, and sources of value, \$ keyword

On OS X and Unix systems, environment variables hold various kinds of information about the operating system. To access an environment variable, you prepend the \$ character to the variable's name.

For example, the following statement displays the current user's login name in a field:

```
put $LOGNAME into field "Login Name"
```

Note: You can create environment variables and change their values using the put command. However, any changes you make to an environment variable are visible only to the application and to any processes it launches, not to other processes. For example, you can't change the \$LOGNAME in the shell.

How to validate the contents of a field

See Also:

How to prevent changing a field's text, How to prevent entering certain characters in a field, How to respond to a change in field contents, How to respond to a keystroke, Recipe for limiting the number of characters in a field, Recipe for rejecting non-digit characters typed into a field, closeField message, exitField message, keyDown message, rawKeyDown message

Certain fields where the user enters data require a particular format. (For example, a ZIP code field might require exactly five digits, with no non-digit characters.) Checking whether the user has entered data in the correct format is called validation. To validate a field's contents, you can take either of two approaches:

ò To check each character the user types, use a keyDown handler. If the key typed is acceptable, pass the keyDown message to allow the character to appear in the field. Otherwise, trapping the message prevents the keystroke from being processed.

ò To validate when the user finishes with the field and moves to the next field (or closes the window, or moves to another card), use a closeField handler. (The closeField message is sent whenever the focus leaves a field, if the field's contents have changed.)

How to write your own commands and functions

See Also:

How to create a code library, How to execute a command, About commands and functions, About messages and the message path, About the structure of a script, commandNames function, function control structure, functionNames function, on control structure

You write custom commands and custom functions by including a handler that defines what the command or function does in a script. Once you have written the handler in a script, you can use the custom command or custom function anywhere that has that script in its message path.

Custom command handlers begin with the on control structure. They may have parameters and are used like any other command.

```
myCustomCommand someVariable -- custom command
```

Custom function handlers begin with the function control structure. They may have parameters, and are always called with parentheses:

```
put myFunction() into field "Result" -- custom function  
get myFunction(the date) -- custom function with 1 param
```

Why am I running out of memory?

See Also:

Why do I get a recursion error?, Why is Revolution taking more memory than its allocation?, Why is there already a stack with the same name?, About Revolution system requirements, Memory and Limits Reference, File menu > Close and Remove from Memory..., File menu > Move Substack to File..., Tools menu > Application Browser, alwaysBuffer property, delete variable command, destroyStack property, filename property, hasMemory function, revUnloadSpeech command, stackFiles property

Revolution reads the entire contents of each open stack file into the application's memory partition. In other words, Revolution's memory must hold not only the application's working space, but all the open stacks and their substacks as well.

If you're running out of memory, try one or more of the following:

- ò Close some open stacks.

- ò Look in the Application Browser window to see whether any closed stack windows are still in memory. (If a stack's destroyStack property is set to false, it remains in memory even after its window is closed.) Control-click a stack (Mac OS or OS X) or right-click (Unix or Windows) and choose "Close and Remove from Memory" from the contextual menu to reclaim the memory that the stack is using.

Note: You can remove a stack from memory only if any other stacks in the same stack file are also closed and removed from memory.

- ò If a stack is large, consider splitting it into several stack files, so that only the part you're working on needs to be in memory at any given time.

- ò If a stack contains large video clip or audio clip objects, or images, consider placing them in separate files and using referenced controls to display them.

- ò If your handlers use large global variables, delete them after you're finished with them to free the memory they use.

- ò On Mac OS systems, increase the application's memory in its Get Info window.

Why are some menu items disabled?

See Also:

Why do menus disappear when I switch cards?, Why does a menu item have missing characters?, Why don't the menus appear when I open a stack?, name property

If a menu item in a menu is unexpectedly disabled, check the following:

• Does the menu item include a parenthesis?

An open parenthesis (in a menu item's text disables that menu item. To include an open parenthesis in a menu item without disabling it, insert a backslash before the open parenthesis: ␣A menu item not disabled␣. The menu item with this text appears in the menu as:

A menu item (not disabled)

• Does your stack's name begin with ␣rev␣?

When a Revolution window (rather than a window you created) is active, certain menu items are disabled to prevent accidentally modifying or deleting part of the development environment.

Since Revolution is itself a Revolution application, every window and dialog box you see in Revolution is a stack. Stack names that begin with the characters ␣rev␣ are reserved by Revolution. Any stack whose name property begins with ␣rev␣ is assumed to be part of the development environment.

If you set the name property of a stack of your own to a name that starts with these three characters, Revolution assumes this stack is part of its development environment and disables certain menu items accordingly. You should avoid setting the name property of your stacks to any name that starts with ␣rev␣.

Tip: If you want a window name in your application to start with ␣rev␣, set the stack's label property to your preferred name, and set the name property to something that doesn't start with those characters.

Why aren't mouse messages sent?

See Also:

Why does a mouseUp handler fail to work?, Why doesn't a control get messages while the mouse is down?, lockMessages property, mouseEnter message, mouseLeave message, mouseMove message, mouseWithin message

When the user clicks or moves the mouse, Revolution sends various messages (mouseMove, mouseDown, mouseUp, and others) to the object under the mouse pointer. If you find that mouse-related messages are not sent correctly, check the following:

- ò Are you are using a tool other than the Browse tool with an editable window? The Revolution development environment traps mouse messages that are sent while a tool other than the Browse tool is being used.
- ò Is the mouse button being pressed? The mouseEnter, mouseLeave, and mouseWithin messages are not sent while the mouse button is down, even if the mouse pointer enters or leaves a control.
- ò Are you clicking a field whose lockText property is false? To make the field receive mouse click messages, either set its lockText to true, or Control-click (Mac OS or OS X systems) or Right-click (Windows or Unix systems).

Why aren't window positions saved in my application?

See Also:

Why can't I save a stack?, Why does a stack window open in the wrong mode?, Recipe for centering a stack window on the screen, location property, rectangle property, windowBoundingRect property

Revolution saves the size and location of each stack window along with the stack file. This means that if your standalone application does not save a stack's file, that stack window's position won't be saved either.

To preserve a stack's window position between sessions of your application, try one of the following:

- ò Place the stack in a stack file (separate from your standalone application). When the window is closed or the user quits the application, save the stack using the save command. The window size and location will be saved along with the stack file.

- ò Save the rectangle property of each window in a preferences file when the user quits the application. When the application starts up, read the preferences file and set the rectangle of each window to the saved size and position.

Why can't I create a standalone Mac OS application?

See Also:

Why can't I save a stack?, Supported Platforms Reference, How to create a standalone application

On Unix and Windows systems, you cannot create a standalone application for Mac OS systems. (You can create OS X, Unix and Windows standalone applications on any system.)

This is because Mac OS applications contain a resource fork, which is a part of the file specific to the Mac OS filesystem and which other filesystems cannot accommodate. To create a standalone application that will run on Mac OS systems, you need to have at least short-term access to a Mac OS or OS X system.

Why can't I display a stack as a normal window?

See Also:

Why does a stack open to the wrong size?, Why does a stack window open in the wrong mode?, Why doesn't Revolution recognize a stack file?, Why don't the menus appear when I open a stack?, How to display another stack in the current window, , modal command, mode property, modeless command, palette command, style property, topLevel command

Revolution calls a normal, editable window a `toplevel` window. By default, stacks open as `toplevel` windows. You can also use the `topLevel` command to change another type of stack window (a palette, modal dialog, or modeless dialog) to a normal editable window.

To change an open stack's mode to `toplevel`, use the Application Browser. Choose Tools menu>Application Browser and Control-click the stack's name (Mac OS or OS X) or right-click (Unix or Windows), then choose `Toplevel` from the contextual menu.

You can also set the mode of a stack by setting its style property. If the style property is set, it overrides attempts to force the stack into a different mode. If opening the stack and using the `topLevel` command doesn't change the window's mode to `toplevel`, try setting the stack's style property to `toplevel`.

Why can't I export an image to a GIF file?

See Also:

export command

The export command supports exporting images in GIF format. However, because of patent issues, exporting a GIF file requires a special license. If you specify the export GIF form and don't have a special license key, the image is exported in Revolution's internal format instead.

If your application needs to export files in this format, contact support@runrev.com to make arrangements for a GIF export key.

Tip: You may want to consider exporting your files in PNG format instead.

Why can't I find a stack I just saved?

See Also:

Why can't I save a stack?, Why can't Revolution find a file I specified?, Why doesn't Revolution recognize a stack file?, How to find out the location of the current stack's file, File menu > Open Recent Stack, File menu > Save, defaultFolder property, mainStack property, name property, save command, substacks property

If you can't find a stack you've saved on your disk, check the following:

ò Are you sure you're looking in the right place? By default, Revolution saves stacks in the same folder as the application, but your stack (or another stack you opened) might have changed the defaultFolder property, causing your stack to be saved elsewhere.

ò Could the stack you saved be a substack of another stack? Revolution can save multiple stacks in the same file. (One stack in a file is the main stack; the rest are substacks of the main stack.) If it's a substack, saving it doesn't create a file for it: substacks are saved in the same file as the substack's main stack.

To see which substacks are in a stack file, open the file, then choose Tools menu>Application Browser and find the name of the main stack. Substacks are listed beneath the main stack.

Why can't I interrupt a handler?

See Also:

How to prevent interrupting a handler, Why does an unwanted handler run?, Shortcut to stop a running handler, `allowInterrupts` property, `cancel` command, `canAbort` property, `pendingMessages` function, `send` command

Normally, you can stop a handler while it's executing by pressing Command-period (on Mac OS or OS X systems) or Control-period (on Unix or Windows systems). This capability can be useful when debugging.

If the key combination does not stop the handler, check the following:

ò Is the stack's `canAbort` property set to true? If the `canAbort` is true, the key combination has no effect on any of that stack's handlers.

ò Is the `allowInterrupts` property set to false? Handlers cannot be interrupted with the key combination if this property is false.

ò Has the handler scheduled a message to run itself again after a period of time? If the handler is interrupted, but has already used the `send` command to schedule another message that causes the same handler to be run again immediately, it may seem as though the handler hasn't been interrupted at all. (You can use the `cancel` command to remove pending messages.)

Why can't I leave all lines of a list field unselected?

See Also:

Why can't I select, copy, or paste text?, Why was the selected text deselected?, `highlightedLine` property, `listBehavior` property, `select` command, `selectedLine` function

When a list field becomes active (focused), if none of its lines is already selected, its first line is selected automatically. Fields become active when the user clicks them, when the user presses the Tab key to navigate from the previous field, or (if the field is the first control on the card) when the card appears.

Occasionally, you might want to create a list field where the first line is not automatically selected. Try the following ideas:

- ò Set the field's `traversalOn` property to false. This prevents the field from becoming active, but if you want the user to be able to select lines by clicking, you must include a `mouseDown` handler in the field to handle the selections. Lines are not automatically selected if the field's `traversalOn` is false.

- ò Create a button behind the list field. On Unix or Windows systems, the control with the lowest layer becomes active when the card opens. By placing another control behind the field, you prevent the field from becoming active when the card opens. Since the field does not become active, its first line is not automatically selected. (This technique does not work if the `lookAndFeel` is set to `Macintosh`.)

Why can't I open a downloaded stack?

See Also:

Why does importing a HyperCard stack cause an error?, Why was a downloaded file corrupted?, How to open a stack, go command

When you double-click a stack file, the operating system must figure out which application the file belongs to in order to open it. Different operating systems use different methods to associate files with their applications, and when you download a stack file—especially one that was created using a different operating system—you may find that this data has been lost in transition, or was never present.

If you have downloaded a stack file and can't open it, try the following:

◦ Was the file compressed? It is common for files that are uploaded to a server to be compressed, in order to decrease download time, and such files must be uncompressed once you download them. Common formats are StuffIt (for Mac OS and OS X systems) and ZIP (for Windows systems). If the file's name has a .sit, .zip, or .gz extension, the file is compressed and needs to be uncompressed before you can use it.

◦ Are you working on a Windows system? The name of a stack file must end in the extension .rev in order to be recognized as belonging to Revolution on Windows systems. Since this extension isn't required on Mac OS systems, a stack file that was created on such a system may lack the extension.

◦ Are you working on a Mac OS system? Mac OS files include a creator signature and file type. Since other operating systems don't support the signature and file type, this data may be missing if you download a stack file that was created on Windows or Unix. Revolution automatically sets the File Exchange control panel to map between the .rev extension and the corresponding signature and file type, so usually the creator signature and file type are set automatically when you download a file. Otherwise, you can use a utility such as ResEdit to restore the proper creator signature (for Revolution files, this is 'Revo') and file type ('RSTK').

◦ Can you open the stack by starting up Revolution and choosing File menu>Open Stack? If you can open the stack, saving it will attach the necessary information for the current operating system so that you can double-click the file to open it.

Tip: To see all files in the Open Stack dialog box, hold down the Option key (Mac OS or OS X) or Alt key (Windows or Unix) while choosing File menu>Open Stack.

◦ Can you open the stack using the go command? Enter the following into the message box:

```
answer file empty;go stack it
```

When the file dialog box appears, choose the stack file.

If the stack file cannot be opened with the previous method, the file is probably corrupted. Try downloading the file again. If you're using FTP to download, make sure to use binary mode, not text mode.

Why can't I resize a control?

See Also:

Why can't I select a control?, Why does an object change size?, height property, lockLocation property, rectangle property, width property

You resize a control by selecting it with the Pointer tool, then dragging one of the small black handles at the sides or corners of the control.

If you can't resize a selected control, check the following:

ò Is the control's lockLocation property set to true? If the lockLocation is true, the handles are gray (instead of black), and the control cannot be resized or moved. Either set the lockLocation to false and then resize the control, or use a handler to change the control's height, width, or rectangle properties instead of changing the size manually.

ò Is the control part of a group? If you try to drag a grouped control outside the group's rectangle, the part of the control that's outside the group's rectangle won't be visible. First select the group and resize it to fit, then drag the control to its new location.

Why can't I save a stack?

See Also:

Why aren't window positions saved in my application?, Why can't I create a standalone Mac OS application?, Why can't I find a stack I just saved?, Why doesn't Revolution recognize my stack file?, Why is there already a stack with the same name?, File menu > Save, File menu > Save As..., defaultFolder property, diskSpace function, mainStack property, name property, save command, substacks property

When you save changes to a stack file, Revolution creates a backup copy before saving, then deletes the backup copy after the file has been saved. This ensures that the data will not be lost if the save action fails. The backup copy has the same name as the file, plus a ~ character at the end of the name.

If saving a stack doesn't work, check the following:

ò If Revolution displays an error message, there may not be enough disk space to create the backup copy, or the operating system may not allow you to create the file in that location, or the name might be too long or contain characters that are not allowed by the operating system, or the ~ backup file may already exist (from a previous save operation that failed). Make sure you can create a file with the same name in the same folder as your stack, that there is enough disk space available, and that a ~ file doesn't already exist in the same folder as the stack you're trying to save.

ò If you successfully saved a stack, but can't find its file, the stack you saved might be a substack of another stack. When you save a substack, no separate file is created for it: substacks are saved in the same file as the substack's main stack.

To see which substacks are in a stack file, open the file, then choose Tools menu>Application Browser and find the name of the main stack. Substacks are listed beneath the main stack.

ò If the stack is a standalone application, it cannot be saved. (On OS X, Unix, and Windows systems, applications can't alter themselves; for consistency, Revolution also imposes this limit on Mac OS standalones.) A standalone application can save stacks in separate files, but cannot save a stack in the standalone's file.

Why can't I select a control?

See Also:

Why can't I resize a control?, Why can't I select a graphic?, Why can't I select, copy, or paste text?, Why doesn't a control appear?, Edit menu > Select All, Edit menu > Select Grouped Controls, Edit menu > Intersected Selections, Tools menu > Pointer Tool, layer property, mouseControl function, select command, selectGroupedControls property

You select a control by clicking it with the Pointer tool. If you can't select a control, check the following:

ò Is the control part of a group? If the control is in a group, make sure Edit menu>Select Grouped Controls is checked, then choose the Pointer tool and click the control.

ò Are you trying to select a group? To select a group, make sure Edit menu>Select Grouped Controls is unchecked, then click any control in the group.

ò Is another control on top of the one you want to select? Try moving the interfering control out of the way temporarily.

ò Is the control's canSelect property set to true?

ò To select a control that's difficult to select manually, enter a select command in the message box.

Why can't I select a graphic?

See Also:

Why can't I select a control?, Why can't I select, copy, or paste text?, Edit menu > Select All, Edit menu > Select Grouped Controls, fill property, ink property, select command

If a graphic's fill property is set to false, its interior is not considered part of the graphic.

To select a graphic whose fill is false, click its border.

To make a graphic easier to select while leaving its interior transparent, set its fill property to true and its ink property to noOp.

Why can't I select, copy, or paste text?

See Also:

Why can't I select a control?, Why can't I use the arrow keys when editing a field?, Why do lines of text overlap?, Why doesn't the Tab key move to the next field?, Why was the selected text deselected?, How to prevent changing a field's text, How to select text in a field, Edit menu > Cut, Edit menu > Copy, Edit menu > Paste, Edit menu > Select All, canModify property, copy command, lockText property, paste command, select command, traversalOn property

You select text in a field using the Browse tool, and copy and paste it using the `Copy` and `Paste` items in the Edit menu. If you can't select, copy, or paste text, check the following:

- ò If the field's `canModify` property is set to true, you cannot paste text (although you can select and copy it).

- ò If the field's `lockText` property is set to true, you cannot change text in it. (However, if the field's `traversalOn` property is set to true, you can select and copy text from it, even if the field is locked.)

- ò If the field's `traversalOn` property is set to false, you cannot select any text in it.

- ò On Unix systems that are set up to use implicit focus, selections are lost when the mouse pointer moves out of the stack window. Since implicit focus causes problems when operating Revolution, you should set up your system to use explicit focus when using Revolution or applications created with it.

Why can't I upload a file?

See Also:

Why don't URLs work in a standalone?, How to create a directory on an FTP server, How to unwedge upload and download operations, How to use a password for a web page, ftp keyword, libURLDownloadToFile command, libURLftpUpload command, libURLftpUploadFile command, libURLSetFTPMode command, load command, put command, URLEncode function

If you can't successfully upload a file, check the following:

ò Are you trying to upload to a web server, with a statement like the following:

```
put it into URL "http://www.example.com/file"
```

Most web servers do not allow you to upload data via HTTP. (Usually, you use the FTP protocol instead for uploading.) Check with the server's administrator to make sure you know the correct protocol and location to use for uploading files.

ò Is the server up and accepting connections, are your user name and password correct, and do you have permission to upload to the directory? Try using another program to upload the same file to the same location, using the same user name and password, to verify whether the problem is with Revolution or not. (Copy and paste the information, if the other program allows it, to make sure you're typing the information correctly.)

ò Does any part of the URL—in particular, the user name or password—contain non-alphanumeric characters? If there are any characters that aren't letters or digits, either the Internet library or the server may become confused and be unable to connect. To avoid this, use URLEncode function to encode any punctuation, special, or control characters.

For example, suppose your user name is `joan` and your password is `still.night`, which contains a punctuation character. The following example shows how to safely encode the password and use the `libURLftpUploadFile` command to upload a file:

```
put "joan" into userName
put "still.night" into userPassword
put "ftp://joan:" & URLEncode("still.night")

& "@ftp.example.org/new/currentStats"

into uploadURL
libURLftpUploadFile theFilePath,uploadURL
```

Use the same precaution, of encoding problem characters in a URL, with any method of uploading or downloading a file that uses a URL.

Why can't I use the arrow keys when editing a field?

See Also:

Why can't I select, copy, or paste text?, Why do lines of text overlap?, Why doesn't the Tab key move to the next field?, How to block or change the action of arrow keys, arrowKey message, navigationArrows property, textArrows property

If the textArrows property is set to false and the navigationArrows property is set to true, the arrow keys move from card to card, rather than moving the insertion point in a field.

Set the textArrows property to true to allow use of the arrow keys in a field. (When the textArrows is true, the setting of the navigationArrows has no effect on field editing.)

You can either check "Arrow keys navigate through cards" in the "Shortcuts" pane of the Preferences window, or enter this statement in the message box:

```
set the textArrows to true
```

Why canÆt Revolution find a file I specified?

See Also:

Why canÆt I find a stack I just saved?, Why doesnÆt Revolution recognize a stack file?, Why donÆt URLs work in a standalone?, About filename specifications and file paths, How to change the current folder used for file operations, How to include a slash in a file path, answer file command, ask file command, defaultFolder property, name property

When specifying a fileÆs location in Transcript, you use either a relative file path or an absolute file path.

ò Make sure that the names of folders in your file path are separated with slashes (/), not colons (:) or backslashes (. To maintain cross-platform compatibility, Revolution uses the Unix-style separator in file paths, regardless of which platform youÆre using.

ò If the file path is a relative file path, make sure the defaultFolder property is set to the location you expect, and has not been changed by your stack or another stack.

ò To obtain the correct file path for a file, enter the following statement in the message box:

answer file "Where is the file?"; put it

This statement displays a dialog box for you to select the file, then places the fileÆs absolute file path in the message box. Study the form of the file path to make sure that file paths you specify in your handlers are correct.

ò On OS X or Windows systems, be sure to include the fileÆs extension. Both these operating systems include a preference setting to hide extensions from the user, but you must use the extension regardless when referring to a file.

ò On OS X systems, many applications (including Revolution applications created with version 2.0 or later) are delivered in the form of application bundles. A bundle is presented to the user as a file, but is actually a folder. If you use an expression like there is a file "MyApp.app", it will evaluate to false, because the application bundle is actually a folder, not a file. Be sure to refer to application bundles as folders, not files.

Why can't Revolution find a handler?

See Also:

Why doesn't a control get messages while the mouse is down?, Why does a mouseUp handler fail to work?, Why isn't a handler executed?, script property, set command

If you get a "Couldn't find handler" error message when sending a message, or using a custom command or custom function, check whether the script was compiled successfully.

When you Apply or close a script in the script editor, Revolution automatically compiles the script, making it ready to run. (Scripts are also compiled when you import a HyperCard stack or when you change the script using the set command.)

If Revolution encounters a compile error while compiling a script, the text of the script is saved but it isn't compiled, and so its handlers cannot be used. You must fix the error and successfully compile the script before any of the handlers in the script can be executed.

To find out whether a script has a compile error, open the script and make a change (such as adding a character and then deleting it), then click Apply. If there is a compile error, the error window appears.

Why do fonts and colors change when I create a standalone application?

See Also:

Why do icons disappear from a standalone application?, Why do objects look different when created in a standalone?, Why does an image have the wrong colors?, Why does text appear distorted?, Why doesn't my application look like other Mac OS applications?, Why don't animations run in my standalone?, Why don't buttons respect my color settings?, About colors and color references, How to change an object's color, How to change the color of text, How to find out the actual color of an object, backgroundColor property, borderColor property, bottomColor property, focusColor property, foregroundColor property, hiliteColor property, shadowColor property, textFont property, textSize property, textStyle property, topColor property

The Revolution development environment sets default values for font and color properties. This means that if your stack's font and color properties are set to empty, these default settings are used to display objects in the stack.

When you start up a standalone application, however, those default settings are not used, since they're part of the development environment rather than the underlying engine. Your stack's objects, therefore, display the default settings for the engine instead of the default settings for the development environment. These settings may differ, depending on platform.

To ensure that your stack's appearance remains the same when you build a standalone application from it, do one of the following:

- ò Set the stack's textFont, textStyle, and textSize properties, and its backgroundColor, foregroundColor, and other color properties, to the values you want before creating the standalone.
- ò On the Stacks tab in Step 3 of the Distribution Builder, make sure the "Copy default font settings" and "Copy default colors" boxes are checked.

These options automatically set any of your stack's font and color properties that you haven't specified to the same settings used by the development environment.

Why do I get a recursion error?

See Also:

Why am I running out of memory?, Why does importing a HyperCard stack cause an error?, lockMessages property, on control structure

When a handler calls itself, it is said to be recursive. Recursion is a powerful programming technique, but when it is unintentional, it can create an infinite regress. If Revolution notices that a recursive handler is creating an infinite regress, it displays an error message.

For example, suppose you have a mouseDown handler that contains this statement:

```
click at the loc of me
```

When you click the object, the mouseDown handler runs. During the mouseDown handler, the click command executes, but the click causes another mouseDown message to be sent, executing another copy of the mouseDown handler, which causes the click command to execute again, which sends another mouseDown message, and so on. The recursion is infinite, and if Revolution didn't halt the recursion, the original mouseDown handler would never finish executing.

Another common source of accidental recursion is using a go command within a closeCard handler (because a closeCard message is sent when the go command executes).

To avoid this sort of accidental infinite recursion, try the following:

- ò Set the lockMessages property to true before executing any command that will cause an infinite recursion. For example, if lockMessages is true, the go command doesn't cause openCard and closeCard messages to be sent, so you can use it safely in an openCard or closeCard handler.

- ò Instead of creating a handler for a system message that's sent automatically, create a custom command and execute it only at the appropriate time. For example, if the click command is contained in a custom handler instead of a mouseDown handler, it can be used safely, since clicking won't execute another copy of the custom handler.

Why do icons disappear from a standalone application?

See Also:

Development menu > Object Library, armedIcon property, disabledIcon property, hilitedIcon property, icon property, showIcon property

Revolution includes built-in libraries of icons, which you access using the Image Library. Once built, a standalone application no longer has access to Revolution libraries, unless you have included the appropriate library when building the standalone. If you create a standalone application that uses any icons from the Standard Images, Classic Images, you must include the correct image library.

To include an icon library, in Step 3 of the Distribution Builder window, click the Inclusions tab and check the "Image Libraries" box. Then select the library or libraries you need from the list below the checkbox.

(This applies only to icons in Revolution's built-in icon libraries. If you use your own custom images for icons, you need not include the icon libraries in your standalone.)

Why do lines of text overlap?

See Also:

Why does an object or window flicker?, Why does text appear distorted?, Why is there a problem with line endings?, Text menu > Size, fixedLineHeight property, formattedHeight property, lineHeight property, textSize property

The lineHeight property of a field determines how much vertical space is allotted for each line of text. If the field's text is too large for the lineHeight, the lines of text may overlap vertically.

To solve this problem, try one or more of the following:

- ò Set the field's fixedLineHeight property to false. This allows the line spacing to adjust to the size of the text in the field, regardless of the lineHeight property.
- ò Set the textSize property of the field to a smaller number. (In other words, reduce the size of the field's text.)
- ò Increase the field's lineHeight property to allow enough vertical space for each line of text.

Why do menu commands affect the wrong stack?

See Also:

Why can't I display a stack as a normal window?, Why do menus disappear when I switch cards?, Why does an unwanted handler run?, Why doesn't Revolution recognize my stack?, cantModify property, defaultStack property, mode property, topStack function

In most applications, menu commands are applied to the active window. Since Revolution gives you the flexibility to use several different window types, not all of which are editable, the current stack is not always the same as the active window.

To determine which stack is the current stack, use the following rules:

1. If any stacks are opened in an editable window, the current stack is the frontmost unlocked stack. (A stack is unlocked if its cantModify property is set to true.)
2. If there are no unlocked stacks open, the current stack is the frontmost locked stack in a normal window.
3. If there are no stacks open in a normal window, the current stack is the frontmost stack in a modeless dialog box.
4. If there are no editable or modeless windows open, the current stack is the frontmost palette.

Another way of expressing this set of rules is to say that the current stack is the frontmost stack with the lowest mode property.

To change the current stack, use the "Default Stack" popup menu in the message box. You can also set the defaultStack property to change the current stack.

Why do menus disappear when I switch cards?

See Also:

Why don't the menus appear when I open a stack?, Why does the top of my stack window disappear when I add a menu bar?, Why does the window change size when I create a card?, About menus and the menu bar, Menu Builder Tutorial, How to show or hide the menu bar, How to switch between menu bars, defaultMenubar property, editMenus property, menubar property

In Revolution, to use a custom menu bar, you create a group of buttons at the top of the stack window to hold the menus. (If you use the Menu Builder to create a menu bar, this is done for you automatically.) On Windows and Unix systems, the menu bar appears at the top of the stack window, and on Mac OS and OS X systems it appears at the top of the screen, in accordance with standard behavior for each platform.

If this menu bar group is not placed on one of the cards in a stack, it will not appear when you go to that card. (This is true even on Mac OS and OS X systems, where the menu bar doesn't appear inside the window.) The group must be placed on all cards where you want the menu bar to be available. For most applications, this will be true on all cards and so you should place the group on all cards of your stack.

If a stack has more than one card and you want the menu bar to appear on each card, place the menu bar group on each card in the stack. (To place the group on a card, choose Object menu>Place Group or use the place command in a handler or the message box.)

Tip: To automatically place the menu bar group on any new cards you create in a stack, set the group's backgroundBehavior property to true.

Tip: On Mac OS and OS X systems, you can set the defaultMenubar property to a menu bar group in any open stack. The specified menu bar appears whenever you're in a stack or on a card that doesn't have a menu bar group of its own.

Why do objects look different when created in a standalone?

See Also:

Why do fonts and colors change when I create a standalone application?, Why do icons disappear from a standalone application?, Why is a custom property garbled when switching platforms?, How to change default settings for new objects, create command, create card command, templateAudioclip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoclip keyword

The Revolution development environment sets default values for the properties of new objects, using the template settings (such as the templateButton for buttons). This means that when you create an object, these default settings, set in the template, are used for the new object.

In a standalone application, those default settings are not used, since they are part of the development environment rather than the underlying engine. Objects you create with the create command display the default settings for the engine instead of the default settings for the development environment. These settings may differ, depending on platform.

To ensure that objects created in your standalone have the property settings you want, set the properties of the appropriate template before you create an object.

For example, to create an opaque graphic, use the following statements:

```
set the opaque of the templateGraphic to true  
create graphic "My Graphic"
```


Why do some characters change when transferred between systems?

See Also:

Why is there a problem with line endings?, Recipe for converting between characters and ASCII values, charSet property, charToNum function, ISOToMac function, macToISO function, platform function

Characters whose ASCII value is less than 128 are the same in the Mac OS character set and the ISO 8859-1 character set used on Unix and Windows systems. These characters include uppercase and lowercase letters, numbers, and most punctuation.

However, characters whose ASCII value is between 128 and 255 are not necessarily the same in both character sets. These special characters are typed with the Option key on Mac OS and OS X, and include characters used for writing languages other than English, accented characters, and some punctuation marks (such as curved quotes).

Because of this, if you create special characters in one character set and display them in the other, the appearance of the characters won't be correct because their ASCII values don't correspond. To translate text that includes these characters, you use the macToISO and ISOToMac functions.

When you move a stack developed on a Mac OS or OS X system to a Windows or Unix system (or vice versa), Revolution automatically translates text in fields and scripts into the appropriate character set. However, in a few situations, you may need to translate text that contains special characters:

ò If you use a text file that was created on a system that uses a different character set, you must translate any special characters in the file. For example, if you are using a Windows system and your application gets a text file from a Mac OS web server, and the file contains special characters such as curved quotes, they must be translated using the macToISO function before you can display them on the Windows system.

ò If you use the read from socket or read from driver command to get text from another system that uses a different character set, you will need to translate the text if it contains special characters.

ò Similarly, if you use the write to socket or write to driver command to send text to another system that uses a different character set, you should translate any special characters before sending the text.

ò If your stack displays data from a SQL database or updates it with data the user enters, translating special characters is necessary if the system that hosts the database does not use the same character set as the system the application is running on.

ò Custom properties are not automatically translated (because if they contain binary data, translating garbles the data). If custom properties in your stack contain text with special characters, you must translate them before using them.

Why do some frames of an animated GIF look strange?

See Also:

`constantMask` property, `currentFrame` property

An animated GIF is created by saving each frame of the image in the same file. When the GIF is played, each frame is displayed, one after the other.

Some programs that create animated GIFs use an optimization trick to reduce the file size: they don't include the changes in the mask between frames. An animated GIF that is encoded in this way will look fine in a program that always plays the GIF from the beginning, but it won't look right if you skip directly to a specific frame.

Since Revolution supports going directly to any frame of an animated GIF, it does not support this optimization trick, and animated GIFs encoded this way may have areas that are invisible or distorted when viewing certain frames in Revolution. If you have an animated GIF with this problem, try one of the following:

- ò Use a GIF editor to re-save the GIF while saving changes to the mask between frames. Most GIF editors have an option to turn off all optimization; this is more than needed, since it requires the program to save each frame of the GIF rather than simply saving the changes between frames, and it will make the finished file larger, but it will solve the mask problem.
- ò Set the image's `constantMask` property to true. (This may cause the image's appearance to be incorrect if you move it while the animation is playing, or if there is an object underneath it.)

Why do unwanted objects appear on new cards?

See Also:

Why does an image disappear?, Why does an object change size?, Why doesn't a control appear?, Object menu > Remove Group, create card command, groupNames property, place command, remove command

When you create a card by using the create card command or by choosing Object menu>New Card, any groups on the current card are automatically added to the new card.

To remove unwanted groups from a card, use the Pointer tool to select the group (making sure that Edit menu>Select Grouped Controls is unchecked) and choose Object menu>Remove Group. You can also use the remove group command to delete unwanted groups from a card, while leaving them on other cards.

Why does a custom cursor fail to appear?

See Also:

Why does an image disappear?, Why doesn't a control appear?, colors property, cursor property, ID property, lockCursor property

If you have created an image to use as a custom cursor, and the cursor does not appear when you set the cursor property to the ID of your image, check the following:

ò Is the image you're using in an available stack? Revolution must be able to find the image in order to use it. One way to check this is to create a button and set its icon property to the cursor image's ID, and set its showIcon to true. If your image does not appear in the button, the image is not available. To make sure the image is available, put it in the stack where you want to use it, or a stack in the same stack file.

ò Is the cursor image the correct size for the platform you're using? On Mac OS systems, custom cursors should be 16x16 pixels. On Unix and Windows systems, custom cursors should be 16x16 or 32x32 pixels.

ò Does the image ID conflict with the ID of another object in the same stack, or with a reserved ID? Conflicting IDs may result in the wrong image being used for the cursor. (IDs below 1000 are reserved for Revolution's built-in resources.)

ò Does the cursor image have the correct number of colors (white, black, and a transparent color)? To find out how many colors the image has, check its colors property. It should contain three lines, one for each color.

If the image has too many colors, the cursor appears as a black rectangle. A cursor you create in another application and import into a stack sometimes is padded with extra colors, even if you have used only three. To reset the number of colors, make a change to the image using Revolution's paint tools. Painting in the image—even changing a single pixel and then changing it back—causes Revolution to correctly store the number of colors.

Why does a handler stop working when moved to another platform?

See Also:

keyword, <> operator, <= operator, >= operator, charSet property

A few characters can be used in Transcript only on Mac OS and OS X systems. (These characters are included for backwards compatibility with imported HyperCard stacks.) On Windows and Unix systems, handlers containing these characters do not function properly.

If you encounter this problem, check the non-functioning handler for the following characters:

ò ¼ (option-L)

Use the synonym instead.

ò _ (option-.)

Use the synonym >= instead.

ò _ (option-,)

Use the synonym <= instead.

ò _ (option-=)

Use the synonym <> instead.

Why does a menu item have missing characters?

See Also:

Why are some menu items disabled?, How to assign a keyboard equivalent to a menu item, How to enable or disable a menu item, About menus and menu bars

Menu items can have several special effects that control the appearance and function of the menu item (disabling it, assigning keyboard shortcuts, and so on). You create these effects by inserting specific characters in the menu item text.

The characters include **&**, **/**, and **(**. If one of these characters appears in a menu item, the character is hidden and the special effect is shown in the menu item instead.

To use one of these characters in a menu item without showing the associated special effect, change it in the menu item text as follows:

Character:	Enter in menu item as:
&	&&
/	//
(

Why does a mouseUp handler fail to work?

See Also:

Why aren't mouse messages sent?, Why can't Revolution find a handler?, Why doesn't a control get messages while the mouse is down?, Why doesn't the controller of a player respond to clicks?, Why isn't a handler executed?, How to simulate a mouse click, click command, mouseDoubleUp message, mouseRelease message, mouseUp message

When the user clicks the mouse, a mouseUp message is sent to the object when the user releases the mouse button. If the mouse is clicked and then released outside the object, or if the mouse is clicked rapidly twice, the mouseRelease or mouseDoubleUp message is sent instead. If you find that the mouseUp message is not sent correctly, try one or more of the following:

ò Add a mouseRelease handler that calls the mouseUp handler:

```
on mouseRelease
    mouseUp
end mouseRelease
```

The mouseRelease message is sent to an object if the user releases the mouse when the mouse pointer is outside the object. Capturing the mouseRelease message prevents the object from missing the mouseUp event if the user clicks, then drags the mouse outside the object's boundaries before releasing it.

ò Add a mouseDoubleUp handler that calls the mouseUp handler:

```
on mouseDoubleUp
    mouseUp
end mouseDoubleUp
```

If the user clicks rapidly twice, the second click sends a mouseDoubleUp message instead of mouseUp. Capturing the mouseDoubleUp message ensures that the object receives a separate mouseUp message for each click.

ò If you are clicking a field whose lockText property is false, the mouseUp message is not sent. (This is because clicking a field normally sets an insertion point, rather than performing another action.) To make the field receive mouseUp messages, either set its lockText to true, or Control-click (Mac OS or OS X systems) or Right-click (Windows or Unix systems).

Why does a repeat loop behave strangely?

See Also:

repeat control structure

If you have a repeat control structure that seems to skip some of the iterations or to end before it's finished processing each iteration, check the structure of the loop to make sure you are not modifying the set to be processed.

For example, suppose you want to delete all the cards in a stack that have a certain word in their name. The obvious way to do this is a repeat loop such as the following:

```
repeat with x = 1 to the number of cards
  if "delete" is in the name of card x
  then delete card x
end repeat
```

Suppose the stack has four cards named A, B-delete, C-delete, and D. We can see from the names that cards 2, and 3 should be removed. Here's what happens when the above repeat loop is executed:

First iteration: $x = 1$.

Card A is not deleted, since its name doesn't contain delete.

Second iteration: $x = 2$.

Card B-delete is deleted.

Third iteration: $x = 3$.

Because the second card was deleted, the third card is now card D. It is not deleted. Card C-delete, which was originally the third card, is never checked, so it isn't deleted.

Fourth iteration: $x = 4$.

Because there are now only three cards in the stack, when the loop tries to check the name of card 4, it causes an execution error.

A repeat control structure that uses the repeat with counter = startValue to endValue form is vulnerable to this problem, if the statements inside the loop delete some of the things that the loop is counting through. (If the statements in the loop don't affect the number of things the loop is counting through, this problem won't occur.)

To fix the problem, change the loop so it goes backward through the set, instead of forward. This prevents the loop from "falling off the end" due to decreasing the number of things. Here's how to rewrite the example above to prevent the problem:

```
repeat with x = the number of cards down to 1
  if "delete" is in the name of card x
  then delete card x
```


end repeat

This example is almost identical to the previous one, except that it uses the repeat...down to form of the repeat control structure. In this case, the loop starts with the last card and goes backward. Here's what happens when this example is executed:

First iteration: $x = 4$.

Card "D" is not deleted, since its name doesn't contain "delete".

Second iteration: $x = 3$.

Card "C-delete" is deleted.

Third iteration: $x = 2$.

Card "B-delete" is deleted.

Fourth iteration: $x = 1$.

Card "A" is not deleted.

The result is that cards 2 and 3 are correctly deleted and the loop works as expected.

Why does a stack open to the wrong size?

See Also:

Why can't I display a stack as a normal window?, Why does a stack open slowly?, How to change the size and position of a window, How to display another stack in the current window, How to put different-sized cards in the same stack, Why does the top of my stack window disappear when I add a menu bar?, height property, resizable property, vScroll property, width property, windowBoundingRect property

If a stack does not open to the size you want, check the following:

ò Does an openStack or preOpenStack handler set the stack's rectangle property (or related properties such as height and width)?

ò Is the windowBoundingRect property set to a rectangle smaller than the saved size of the stack? By default, Revolution sets this property to leave room for the Toolbar at the top of the screen, and for platform-specific elements such as the Windows toolbar. You can set this property to another value. You can also set the stack's rectangle property in a preOpenStack handler to prevent its being constrained to the windowBoundingRect.

ò On Mac OS and OS X systems, is the stack's menubar property set to a group in the stack? When you set the menubar property to a group of buttons, the buttons are displayed as menus in the menu bar. On every card that contains the group, the stack window is shortened and scrolled up so that the group is hidden. (To reverse this action so you can see, select, and edit the buttons that make up your menu bar, set the editMenus property to true.)

Why does a stack window open in the wrong mode?

See Also:

Why aren't window positions saved in my application?, Why can't I display a stack as an editable window?, Why does a stack open to the wrong size?, Why doesn't Revolution recognize a stack file?, Why don't the menus appear when I open a stack?, About windows, palettes, and dialog boxes, modal command, mode property, modeless command, palette command, style property, topLevel command

If a stack's style property is set to a specific mode, the property setting overrides attempts to force the stack into a different mode using the topLevel, palette, modal, or modeless commands.

If opening a stack and using one of these commands doesn't change the window's mode to the one you want, try setting the stack's style property to the desired window mode.

Why does a variable lose its value?

See Also:

Why am I running out of memory?, How to declare a variable, constant command, global command, local command, put command, variableNames function

When you use a variable without declaring it first, Revolution creates it as a local variable. Local variables exist only as long as the current handler is running, and disappear as soon as the handler exits. If you execute the handler again, the variable does not retain its old value.

If you declare a variable as a local variable in a script, but outside any handler, the variable can be used by any handler in that script, and retains its value between handlers and between executions of the same handler. However, if the script is recompiled, the local variable loses its value.

If you want a variable to retain its value for the entire session, even if you change a script, use the global command to declare it as a global variable. Global variables continue to exist, and retain their values, until you quit the application.

All variables, whether local or global, cease to exist when you quit the application. If you need to retain a value between sessions, store it in a stack—either in a custom property of an object, or in a fieldÆs contents. You can also store a value in a text file and read the file when you need the value.

Why does an image disappear?

See Also:

Why doesn't a control appear?, Why does an object or window flicker?, Edit menu > Clear, filename property, hide command, platform function, show command, there is a operator, visible property

If an image that is supposed to appear in a stack disappears, check the following:

ò Is the picture stored in another file and displayed in a referenced control? Check the image file to make sure its name or location hasn't been changed. Also check the image's filename property to make sure the file path is correct.

ò Has the stack been moved to another system? If the image is used as an icon or cursor, it may actually reside in another stack that was not moved. If the image is displayed in a referenced control, you may need to change its filename property.

ò Is the image in PICT format? Revolution cannot display PICT images on a Unix or Windows system. If an image was imported from a PICT file and you want it to work on Unix or Windows systems, you must save it in another format such as GIF or JPEG, then re-import it into the stack, in order to make it cross-platform compatible.

ò Is the image actually an EPS object? EPS objects are not displayed on Mac OS, OS X, or Windows systems, or on Unix systems that don't have Display PostScript installed.

Why does an image have the wrong colors?

See Also:

Why do fonts and colors change when I create a standalone application?, Why does an image disappear?, Why does an object or window flicker?, Why is there a border around controls?, About colors and color references, colorMap property, ink property, lockColormap property, privateColors property, screenColors function, screenDepth function

If your system uses 8-bit video (256 colors) or lower, and some of the colors in your stack aren't in the available 256 colors,

If the lockColorMap property is true, Revolution maps each pixel to the nearest available color. This may cause your images to look washed out or mis-colored, and may cause visible bands of color to appear in images that use many similar shades (such as photographs).

If your images look bad on systems using 8-bit video or lower, try one or more of the following:

- ò Set the lockColorMap property to false. This allows Revolution to automatically update the color table when an image is displayed, reducing banding or other problems.
- ò Use the colorMap property to make the system use a custom color table. You can create a set of colors that's specifically optimized for your images.
- ò Use an image-editing program to remap all the images to use the same set of colors. The images will still be reduced to no more than 256 colors, but doing this in an image-editing program gives you more control over the final result than relying on Revolution's automatic color remapping.

Why does an object change size?

See Also:

Why can't I resize a control?, Why does a stack open to the wrong size?, Why does an object or window flicker?, Why does the top of my stack window disappear when I add a menu bar?, Why is a control the wrong size when created by a handler?, Geometry Management Tutorial, How to determine the dimensions of an image, height property, lockLocation property, rectangle property, width property

Normally, objects retain the size you give them. If an object's size changes unexpectedly, check the following:

ò Has a handler changed the object's size? The size can change if the handler alters the object's height, width, or rectangle properties.

ò Is the object an image or player? These controls automatically resize themselves to fit the picture or movie, whenever the card the object is on appears. If you want to set the size of an image or player to a size different from the size of the picture or movie it contains, set the object's lockLocation property to true. This prevents automatic resizing.

ò Is the object a group? Groups automatically resize themselves to fit all the controls in the group. To stop a group from automatically resizing itself, set the group's lockLocation property to true.

Why does an object or window flicker?

See Also:

Why can't I resize a control?, Why does an image disappear?, Why does an image have the wrong colors?, Why does my stack open slowly?, How to prevent displaying intermediate changes on a card, alwaysBuffer property, bufferHiddenImages property, layer property, lockColorMap property, lockScreen property, screenNoPmaps property

If a window, or an object in the window, flickers when you perform certain actions (like selecting text, moving from card to card, or moving controls around), try the following:

- ò Set the stack's alwaysBuffer property to true. If a stack's alwaysBuffer is false, controls are drawn directly on the screen, which may cause flickering when you move a control or change cards. Setting the stack's alwaysBuffer to true eliminates this source of flickering, at the cost of using more memory.

- ò If a player or image flickers, set its alwaysBuffer property to true. If the alwaysBuffer of a player or image is false, it may flash when being drawn on the screen. Setting the alwaysBuffer property to true eliminates this flashing (again, at the cost of greater memory use).

- ò Lock the screen with the lock screen command during execution of a handler that moves, hides, shows, or otherwise manipulates objects. Locking the screen prevents transient flashing as objects are moved or changed.

- ò Does an image or EPS object overlap a field you're typing in? If an image or EPS object overlaps a field, it is redrawn after every character typed into the field. To eliminate this problem, change the layer property of the image or EPS object so it's underneath the field, or move it so it does not overlap.

Why does an unwanted handler run?

See Also:

Why can't I interrupt a handler?, Why do I get a recursion error?, Why do unwanted objects appear on new cards?, Why does a mouseUp handler fail to work?, How to temporarily remove a portion of a script, How to throw away unwanted mouse clicks, How to trigger a handler, exit command, me keyword, target function

When you perform an action such as clicking, the corresponding message or messages are sent to the target object. If that object's script doesn't contain a handler for the message, the message is passed to the next object in the message path. If that object has a handler for the message, the handler is executed; otherwise, the message is passed to the next object in the message path.

If this leads to unexpected results, causing a handler to be executed when you don't want it to be, try one or more of the following:

ò Place an empty handler in the object's script to trap the message. For example, to trap an openStack message sent to a substack and prevent it from being passed to the main stack, place the following handler in the substack's script:

```
on openStack
    exit openStack
end openStack
```

ò To prevent a main stack from getting a message intended for another stack, move the substack to a separate file. This makes it a main stack and removes the main stack from its message path.

ò Check the target function in the offending handler, and use an exit statement to exit the handler if the target is the object that first received the message is not an intended recipient.

Why does importing a HyperCard stack cause an error?

See Also:

About Revolution for experienced HyperCard developers, How to import a HyperCard stack, HCAddressing property, HCIImportStat property, HCStack property

When you open a HyperCard stack from within Revolution, the stack is automatically converted into a Revolution stack. Almost all HyperCard stacks can be imported successfully. If you have problems importing a HyperCard stack, check the following:

- ò If the HyperCard stack does not appear in the Open dialog box, make sure its file type is `ôSTAKö`. HyperCard sets the correct file type automatically when it saves a stack, but if a HyperCard file has been transmitted over the Internet, it may have lost its file type.
- ò If an error message appears when you try to open the HyperCard stack, make sure the stack has been compacted. To compact a HyperCard stack, open it in HyperCard and choose `ôCompact Stackö` from HyperCard's File menu. Then try opening the stack in Revolution.
- ò If the stack opens but a script error message appears, usually this indicates that the script uses one of the few HyperTalk language elements that is not compatible with Transcript. Click the `ôScriptö` button in the error window and comment out the offending line of code temporarily. Check the Transcript Dictionary to find out whether the command, function, or property requires different syntax in Transcript.

Almost all HyperTalk code also runs as Transcript. However, there are a few areas where your old HyperTalk code may be incompatible with Revolution. In this case, as Revolution attempts to compile the script, an error message appears. Depending on exactly what's in your HyperCard stack, several of these errors may appear during the import process.

These are some common sources of script errors in imported HyperCard stacks:

- ò The `doMenu` command is not implemented by Revolution for all HyperCard menu items. A `doMenu` command that specifies a menu item that doesn't exist in Revolution will cause a script error.
- ò Externals (XCMDs and XFCNs) are not automatically imported, so calls that use them cause a script error.
- ò HyperCard's color architecture relies on the `AddColor` external and does not work in Revolution. In Revolution, you specify the color of an object using properties such as `foregroundColor` and `backgroundColor`.

Why does my application quit when I close its windows?

See Also:

File menu > Quit, decorations property

Because the menu bar on Mac OS and OS X systems is at the top of the screen rather than inside a window, it is common for applications to continue running after all their windows are closed, because the menu bar is still present even if no windows are open.

Revolution applications, however, cannot continue running after their last window is closed. If you close all windows in a Revolution application, the application quits.

If you want your Revolution application to continue running even if the user closes all windows, consider including one of the following in your application's design:

- ò Keep one window open all the time, but hide it when it is not in use. If a window is open but hidden, Revolution will not automatically quit. (The application's splash screen can be used for this purpose.)
- ò Keep a settings palette, a fixed button bar, or other window open at all times. Even though a palette is not a document window, if it is open, Revolution will not automatically quit. To make sure the user can't close such a window, set its decorations property to hide its title bar or close box.

Why does my stack open slowly?

See Also:

Why am I running out of memory?, Why can't I display a stack as a normal window?, Why does a stack open to the wrong size?, Why is a custom property garbled when switching platforms?, File menu > Open Stack..., charSet property, openBackground message, openCard message, openStack message, password property, preOpenStack message

If a stack is taking an unusually long time to open, check the following:

ò Does the stack have a time-consuming preOpenStack handler? If this handler takes a long time to execute, consider moving parts of it to an openStack handler (which runs after the stack appears), or structuring it in such a way that not everything needs to be done when the stack opens.

ò Is the stack file a large one? Revolution reads the entire contents of each stack file into the application's memory partition when the stack opens. If a stack is large, consider splitting it into several stack files, so that only the part you're working on needs to be in memory at any given time. If a stack contains large video clip or audio clip objects, or images, consider placing them in separate files and using referenced controls to display them, so that they don't need to be loaded into memory until they're displayed.

ò Is the stack password-protected? Revolution decrypts a password-protected stack every time it opens, and the process takes a few seconds if the stack is large.

ò Is a stack that was last saved on a Mac OS or OS X system being opened on a Unix or Windows system, or vice versa? When you save a stack, its text is encoded using either the ISO character set (on Unix or Windows systems) or the Macintosh character set (on Mac OS or OS X systems). If you open the stack on a system that uses the other character set, Revolution converts all the text in the stack to use the current character set, and the process takes noticeable time if the stack contains a great deal of text in fields or scripts.

Why does Revolution ask to purge a stack?

See Also:

Why is there already a stack with the same name?, About windows, palettes, and dialog boxes, How to find out whether a window is open, openStacks function, reloadStack message, revLoadedStacks function

Revolution cannot have more than one stack with the same name in memory at the same time. (Remember that a stack can be loaded into memory even if its window is not open.)

Because of this, if you try to load a stack that has the same name as a stack already in memory, Revolution asks what you want to do with the existing stack before opening the new one. You can:

- ò Save the existing stack and close it, then open the new stack.
- ò Purge (remove) the old stack from memory without saving it, then open the new stack.
- ò Cancel opening the new stack. The old stack remains in memory, and the new stack is not opened.

The most common way to trigger this message is to open a stack with the same name as one thatÆs already open. It may also be triggered when you get a property of a closed stack, search a closed stack with the Find dialog box, or do anything else that refers to the objects in a stack. This is because Revolution must load the stack into memory in order to access any of its contents.

Tip: If you need to access more than one stack with the same name at the same time, temporarily change the name property of the open stack.

Why does text appear distorted?

See Also:

Why do fonts and colors change when I create a standalone application?, Why do lines of text overlap?, Why is some text blue and underlined?, Why is there a problem with line endings?, Text menu > Font, fontNames function, formatForPrinting property, textFont property

If the text in your stack appears distorted or abnormal, check the following:

ò Is the formatForPrinting property set to true? This property should be set to true only during actual printing. Setting it to true for screen display changes the display of fonts and may produce unexpected results.

ò Are the correct fonts installed on your system? If you create a stack that uses a certain font, and then either remove that font or move the stack to a system where the font isn't available, Revolution substitutes another font. Depending on the differences between the original and substituted fonts, the text may appear distorted.

Why does the Geometry pane move objects to the wrong place?

See Also:

Why doesn't a control appear?, Geometry Management Tutorial, `revCacheGeometry` command, `revUpdateGeometry` command

If you've used the Geometry pane in the property inspector to specify how controls are moved and resized when the user resizes the stack window, and you find that resizing the window causes objects to be moved to the wrong place, the geometry settings may need to be re-cached.

When the Geometry pane moves and resizes an object, it calculates the object's new location and size starting from its baseline location and size. This baseline is stored when you set the object's scaling or positioning settings.

If you move or resize the object after setting up its Geometry settings, you may need to update the stored baseline location and size. This is done using the `revCacheGeometry` command.

To update the settings, use the following statement in a handler or the message box, after you change an object's size or position but before changing cards or resizing the window:

```
revCacheGeometry
```

Why does the top of my stack window disappear when I add a menu bar?

See Also:

Why does a stack open to the wrong size?, Why does an object change size?, Why does the window change size when I create a card?, Why do menus disappear when I switch cards?, Why don't the menus appear when I open a stack?, About menus and the menu bar, Tools menu > Menu Builder, defaultMenuBar property, editMenus property, menubar property

In Revolution, you create a menu bar that works on all platforms by creating a button for each menu, then making those button menus into a group that's placed at the top of the stack window. This is the standard location for the menu bar on Unix and Windows systems.

To create a Mac OS menu bar, you create the group, then set the menubar property of your stack to the name of the group. This does two things: it displays the menus in the menu bar at the top of the screen, and it shortens the stack window and scrolls it up so that the group of menu buttons is not visible in the window. (Since the menus are in the menu bar, you don't need to see them in the stack window as well.)

This ensures that the same menu bar appears in the correct, standard location for each platform. On Windows and Unix systems, the menu bar appears at the top of the stack window as a group of button menus; on Mac OS and OS X systems, the menu bar appears at the top of the screen, and the window is adjusted so the button menus aren't visible.

The Menu Builder takes care of these steps automatically when you check "Set as menu bar on Mac OS". When this option is turned on, the menus are created as buttons across the top of the stack window, and (on Mac OS and OS X systems) the window size is adjusted so the buttons can't be seen. If any of your controls were near the top of the window—if they overlap the strip across the top where the menus are placed—they'll be partially hidden when the stack window is scrolled up.

To revert to the original size and access the area including the menus, set the stack's editMenus property to true. You can then adjust the position of any controls so they don't overlap the menus. Finally, set the editMenus back to false to prevent seeing the menu bar at the top of the stack window.

Why does the window change size when I create a card?

See Also:

About menus and the menu bar, Tools menu > Menu Builder, Why does a stack open to the wrong size?, Why does an object change size?, Why does the top of my stack window disappear when I add a menu bar?, Why do menus disappear when I switch cards?, Why don't the menus appear when I open a stack?, defaultMenubar property, editMenus property, menubar property

In Revolution, you create a menu bar by creating a group of buttons that's placed at the top of the stack window. To create a Mac OS or OS X menu bar, you create the group, then set the menubar property of your stack to the name of the group. This does two things: it displays the menus in the menu bar at the top of the screen, and it shortens the stack window and scrolls it up so that the group of menu buttons is not visible in the window. (Since the menus are in the menu bar, you don't need to see them in the stack window as well.)

If the current card has a menu bar group whose backgroundBehavior property is false, creating a new card does not automatically place the menu bar group on the new card. Because the new card does not include the menu bar group, the stack window does not need to be shortened and scrolled up to accommodate it. When you go from a card that includes the menu bar group to one that doesn't, therefore, the stack window gets taller. When you go back to a card that includes the menu bar group, the stack window gets shorter.

To fix this problem, make sure the menu bar group is placed on all the cards in the stack. To automatically place the group on new cards, set its backgroundBehavior property to true. To place the menu bar group on cards you've already created, choose Object menu>Place Group, or use the place command.

Tip: To revert to the original size and access the area including the menus, set the stack's editMenus property to true.

Why doesn't a control appear?

See Also:

Why can't I select a control?, Why does an image disappear?, Why does the Geometry pane move objects to the wrong place?, Why don't the menus appear when I open a stack?, View menu > Show Invisibles, location property, lockLocation property, rectangle property, there is a operator, visible property

If a control does not appear in the stack window, try the following:

ò To check whether the control exists, use the there is a operator. For example, to check whether there is a button named "Options" in the current stack, enter the following expression into the message box:

```
put (there is a button "Options")
```

ò If the control exists, use the show command to display it.

ò If the control still does not appear, check its location property to make sure it's within the rectangle of the stack window.

ò If the control is part of a group, the control may be outside the group's rectangle, even if it is within the visible area of the stack window.

If a group's lockLocation property is false, the group automatically resizes itself to display all visible controls in the group whenever the group is displayed. To force a group to resize itself to include a newly-shown control, set the group's lockLocation property to false.

Why doesn't a control get messages while the mouse is down?

See Also:

Why aren't mouse messages sent?, Why does a mouseUp handler fail to work?, About messages and the message path, lockMessages property, mouse function, mouseMove message, send command

While the mouse button is being held down, mouseEnter and mouseLeave messages are not sent. This can cause problems if your stack relies on these messages to update the cursor, a toolbar, or other parts of your application.

To make a control react to the mouse's motion even while the mouse button is being held down, try one of the following:

ò Use a mouseMove handler to check the position of the mouse pointer. The mouseMove message is sent every time the pointer is moved, and it continues to be sent while the mouse button is being held down. The following example shows how to check whether the pointer has moved outside the control:

```
on mouseMove
  if the mouseLoc is not within the rect of me
    then send "theMouseHasLeftTheBuilding" to me
end mouseMove
```

(Since the mouseMove message will continue to be sent while the mouse is down, if you use the above method, you may want to set a variable to keep track of whether the mouse has left the control, and execute the handler only if the variable hasn't been set yet. This avoids repeatedly executing the handler every time the mouse moves.)

ò Within a mouseDown handler, use a repeat control structure that continually checks the mouse's position, and performs appropriate tasks if the mouseLoc has changed. This approach is simple and direct, but performance is significantly affected when a handler repeatedly queries the state of the mouse.

ò Use a repeat control structure to send a periodic message, which triggers another handler to perform appropriate tasks.

```
on mouseDown
  repeat until the mouse is up
    send "testMouse" to me in 6 ticks
    wait 6 ticks
  end repeat
end mouseDown
```

The repeat control structure continues to execute as long as the mouse is held down, sending a `testMouse` message every tenth of a second; the testMouse handler can check the mouse's position and perform the appropriate tasks.

This method is preferable to the first because the mouse status is checked only when it needs to be, rather than being checked continuously. Continuous checking of the mouse status can affect performance.

Why doesn't a custom property appear in the property inspector?

See Also:

Why is a custom property garbled when switching platforms?, About custom properties and custom property sets, How to find out whether a custom property exists, How to list an object's custom properties, Object menu > Object Inspector, Object menu > Card Inspector, Object menu > Stack Inspector, customKeys property, customPropertySets property

Revolution maintains custom properties for each object to handle aspects of the object's appearance and behavior. The names of these properties and property sets begin with the characters `ôrevö`. Names that begin with these characters are reserved by Revolution, and any property whose name begins with `ôrevö` is assumed to be part of the development environment.

To display all custom properties—including those reserved by Revolution—choose View menu>Revolution UI Elements in Lists to check it.

Why doesn't a property appear in a profile?

See Also:

When you add a property to a property profile, in the property inspector, the Add Property dialog box lists all applicable properties, but the Property Profiles pane automatically eliminates redundant properties and property synonyms.

For example, if you add the `backgroundColor` and `foregroundColor` properties to a profile, the Property Profiles pane displays the `colors` property the next time you re-open the property inspector. This is because the `colors` property contains all eight color settings of an object, so it's not necessary to store the individual color properties once they've been set.

Similarly, you can add any or all of the `left`, `top`, `right`, `bottom`, `topLeft`, `bottomRight`, and `location` properties to a profile. But since all these properties can be derived from the object's `rectangle` property, the Property Profiles pane stores only the `rectangle` and not the derivative properties.

Why doesn't a stack maximize or zoom to the full screen size?

See Also:

Why does a stack open to the wrong size?, height property, resizable property, width property, windowBoundingRect property

When the user zooms (on Mac OS or OS X systems) or maximizes (on Unix or Windows systems) a stack window, the size of the window is constrained by the windowBoundingRect property.

By default, Revolution sets this property to leave room for the Toolbar at the top of the screen, and for platform-specific elements such as the Windows toolbar. You can set this property to another value, or resize the stack manually (or by changing its rectangle or related properties).

Why doesn't an EPS object appear?

See Also:

EPS objects are supported only on Unix systems that support Display PostScript.

If you attempt to create an EPS object on a system that does not support Display PostScript, or if you open a stack containing an EPS object on such a system, the EPS object appears as a blank rectangle on the card, and the PostScript output is not displayed.

Why doesn't my application look like other Mac OS applications?

See Also:

Why do fonts and colors change when I create a standalone application?, Supported Platforms Reference, lookAndFeel property, proportionalThumbs property

On Mac OS and OS X systems, applications use built-in system routines to draw windows and their contents. These routines reflect the user's system-wide preferences.

If the lookAndFeel is set to `Macintosh`, Revolution uses its own customized routines rather than the standard routines to draw controls. These customized routines mimic the default Mac OS Platinum appearance. This means buttons, scrollbars, popup menus, and other parts of the user interface do not change to match the system appearance when you change them in a control panel.

To make your application reflect changes in the user's settings for fonts, colors, and general appearance, set the lookAndFeel to `Appearance Manager`.

Why doesn't Revolution recognize a stack file?

See Also:

Why can't I display a stack as a normal window?, Why can't I save a stack?, Why can't Revolution find a file I specified?, Why is there already a stack with the same name?, File menu > Open Stack..., File menu > Open Recent Stack, answer file command, ask file command, save command, stackFileType property

Since different operating systems use different methods to associate documents with applications, when you move a stack file from one system to another, Revolution may fail to recognize the file. It may even appear to be a plain text file.

If you encounter this problem, try the following:

ò If you are moving a stack from a Mac OS system to an OS X, Unix, or Windows system, change the stack file's name to end with `.revö`. For example, change the file `My Stackö` to `MyStack.revö`.

ò If you are moving a stack file from a Unix or Windows system to a Mac OS or OS X system, use ResEdit, Snitch, or a similar utility to change the file's type signature to `öRSTKö` and its creator signature to `öRevoö`.

ò If you frequently download or receive stacks on a Mac OS system, use the File Exchange control panel to add a PC Exchange mapping, with `örevö` as the extension and Revolution as the application to map it to. File Exchange will automatically set Revolution files you receive to have the correct creator and type signature.

ò If the above methods don't work, or you simply want to quickly open a stack, enter the following into the message box:

answer file "Choose a stack:"; go to stack it

Once your stack is open, choose File menu>Save As to create a copy of the stack with the correct extension or creator and type signature.

Why doesn't the ask or answer dialog appear in my standalone?

See Also:

Why do fonts and colors change when I create a standalone application?, Why does an image disappear?, Why does a custom cursor fail to appear?, Why don't animations run in my standalone?, answer command, ask command

The dialog boxes used by the ask and answer commands, as well as other dialog boxes, are part of Revolution's development environment and are not automatically included when you create a standalone application. Once built, a standalone application no longer has access to Revolution's internal resources. Unless you include the necessary dialog boxes, these commands will not work properly in your standalone.

To include these dialog boxes (and other resources), make sure the appropriate options in the Distribution Builder are checked when you create your standalone, before you click Build Distribution. The checkboxes for these items can be found on the Inclusion tab of Step 3 in the Distribution Builder.

Why doesn't the controller of a player respond to clicks?

See Also:

Why can't I select a control?, Why does an object or window flicker?, Why don't movies play?, alwaysBuffer property, currentTime property, playLoudness property, showController property, start command, stop command

If the alwaysBuffer property of a player is set to true, the player's controller bar does not respond to clicks.

To solve this problem, either set the player's alwaysBuffer property to false, or hide the controller bar and script your own buttons to control the player's actions.

Why doesn't the Tab key move to the next field?

See Also:

Why can't I select, copy, or paste text?, Why doesn't the Tab key move to the next tab stop?, Why is there a border around controls?, Why was the selected text deselected?, How to change the tab order of controls on a card, lockText property, lookAndFeel property, tabGroupBehavior property, tabStops property, traversalOn property

If pressing the Tab key does not move the insertion point to the next field, try one or more of the following:

- ò If pressing the Tab key moves the insertion point within the field, try setting the field's tabStops property to empty.

- ò If neither pressing the Tab key nor clicking the field moves the insertion point to the field, make sure the field's lockText property is set to false and its traversalOn is set to true.

- ò If pressing the Tab key does not move the insertion point to a field, but clicking the field does, make sure the traversalOn property of other controls on the card is set to false.

If the lookAndFeel property is set to `ôMotifö` or `ôWindows 95ö` (Unix or Windows systems), the Tab key moves the focus to the next control whose traversalOn is true, whether it's a field or not.

- ò If the field is part of a group, and pressing the Tab key skips the other fields in the group, make sure the group's tabGroupBehavior property is set to false.

- ò If pressing the Tab key moves to another field, but not the one you want it to move to, change the layer property of the fields so that the first field has the lowest layer and each successive field has a higher layer. The tab order is the same as the layering order.

Why doesn't the Tab key move to the next tab stop?

See Also:

Why can't I select, copy, or paste text?, Why doesn't the Tab key move to the next field?, How to change the tab order of controls on a card, `tabGroupBehavior` property, `tabStops` property, `traversalOn` property

If pressing the Tab key does not move the insertion point to the next tab stop in the field, try one or more of the following:

- ò Make sure the field's `tabStops` property is set. (If the `tabStops` is empty, the field has no tab stops, and pressing the Tab key moves the insertion point to the next field.)

- ò If the field is part of a group, make sure the group's `tabGroupBehavior` property is set to false. If a group's `tabGroupBehavior` is true, you cannot press the Tab key to go to the next tab stop (as set in the `tabStops` property) in any of the fields in the group. Pressing the Tab key moves to the next control in the group instead.

Why don't animations run in my standalone?

See Also:

Why do fonts and colors change when I create a standalone application?, Animation Builder Tutorial, revPlayAnimation command, revStopAnimation command

Revolution contains a library of custom commands used by the Animation Builder to run animations. Once built, a standalone application no longer has access to Revolution libraries, unless you have included the appropriate library when building the standalone. If you create a standalone application that uses the revPlayAnimation, revStopAnimation, or revGoToFramePaused command, you must include the Animation library.

To include this Revolution custom library, in Step 3 of the the Distribution Builder window, make sure the "Animation Library" option on the Inclusions tab is checked.

(This applies only to animations you create with the Animation Builder. If you use another technique for animation—for example, if you use a QuickTime movie in a player or video clip—you need not include the custom Animation library in your standalone.)

Why don't buttons respect my color settings?

See Also:

Why do fonts and colors change when I create a standalone application?, How to change an object's color, View menu > Look and Feel, colors property, lookAndFeel property, style property

On Mac OS and OS X systems, when the lookAndFeel property is set to `Appearance Manager`, most standard controls are drawn by the operating system's Appearance Manager routines, rather than by Revolution. This means that those routines control the button appearance, so most of Revolution's color and pattern properties have no effect on these control types when the lookAndFeel is set to `Appearance Manager`:

- Option menus and combo boxes: The only color property that affects the button's appearance is the `foregroundColor`.

- Checkboxes and radio buttons: The color properties do not affect the actual check box or circle, but affect the rest of the button.

- Standard buttons and rectangle buttons: If the button's `backgroundColor` and `backgroundPattern` properties (and the `backgroundColor` and `backgroundPattern` of all its owners) are empty, the button is drawn by the operating system and the only color property that affects the button's appearance is the `foregroundColor`. If the `backgroundColor` or `backgroundPattern` is set, all color properties affect the button.

- Other button styles: The color properties work normally regardless of the lookAndFeel setting, since these button types are not drawn by the operating system.

If you want these button styles to display the colors you set, you can set the lookAndFeel to `Macintosh`. This will cause the button to be drawn by Revolution instead of by the operating system routines, allowing you to color it. However, it causes all controls to be drawn with the Macintosh Platinum appearance, which may make your application look out of place if it is running on an OS X system or on a Mac OS system using a non-standard theme.

Why don't movies play?

See Also:

Why doesn't the controller of a player respond to clicks?, Why don't movies work on a Windows CD-ROM?, How to find out whether QuickTime is available, How to play a streaming QuickTime file, platform function, play command, videoClipPlayer property

If the movie does not appear in the player, check the movie file to make sure its name or location hasn't been changed. Also check the player's filename property to make sure the file path is correct.

If a player's alwaysBuffer property is set to true, the player doesn't respond to clicks in its controller bar. To play such a movie, use the start, stop, and play commands.

On Windows systems, conflicts between the graphics card and driver and QuickTime may cause movies to fail to play or to be distorted. If this occurs, try opening the QuickTime control panel and checking the "Safe Mode (GDI Only)" option in the Video Settings section. Also check whether the movie plays successfully in the Windows QuickTime Player.

On Unix systems, Revolution uses the utility "xanim" to play movies. If movies don't play properly on your Unix system, check the following:

- Is the latest version of xanim installed?

- If more than one copy of xanim is installed on your system, does the latest version come first in your PATH? To check which copy of xanim Revolution will use for movies, use the command "which xanim" in your Unix shell.

- Do xanim's error messages tell you more? If you run xanim from the shell (instead of using Revolution to play the movie), you'll see error messages that may tell you what's wrong. Running xanim from the shell can help you in debugging problems with movie playing.

Why don't movies work on a Windows CD-ROM?

See Also:

Why don't movies play?, How to find out whether QuickTime is available, platform function

If your application is placed on the top level of a CD-ROM disc, some versions of Windows fail to correctly return path information. The effect on your application is that players, which reference a file on the CD-ROM, do not work correctly, because the file can't be found.

To prevent this problem from appearing on Windows systems, if you create a CD-ROM for your application, always place the application file in a folder on the disc. As long as the application is not on the top level of the CD-ROM, this problem does not occur.

Why don't the menus appear when I open a stack?

See Also:

Why are some menu items disabled?, Why does the top of my stack window disappear when I add a menu bar?, Why do menu commands affect the wrong stack?, Why do menus disappear when I switch cards?, About menus and the menu bar, How to show or hide the menu bar, Development menu > Suspend Development Tools, Shortcut to open a stack without the development environment, environment function, defaultMenubar property, menubar property

If you hold down the Command key (Mac OS and OS X) or Control key (Unix and Windows) while double-clicking a stack to launch Revolution, the stack opens without the Revolution development environment being loaded.

The Revolution development environment includes the standard menus, the Toolbar, and palettes you use to develop stacks, so if the development environment is not loaded, these tools do not appear and you see only whatever menus and windows are provided by your stack.

Why don't URLs work in a standalone?

See Also:

Why can't Revolution find a file I specified?, Why can't I upload a file?, How to access the Internet from behind a firewall, ftp keyword, http keyword, URL keyword

Revolution contains a library used to access ftp and http URLs. Once built, a standalone application no longer has access to Revolution libraries, unless you have included the appropriate library when building the standalone. If you create a standalone application that uses the http or ftp URL types, you must include the Internet library.

To include this Revolution custom library, in Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked.

Why don't visual effects work?

See Also:

How to find out whether QuickTime is available, dontUseQT property, dontUseQTEffects property, hide command, show command, unlock screen command, visual effect command

Visual effects are used with the visual effect, hide, show, and unlock screen commands. By default, Revolution uses QuickTime to produce these transition effects on Mac OS, OS X, and Windows systems. QuickTime provides a broad range of effects and settings that can be used with these commands.

However, slower machines may not be powerful enough to produce good results when using QuickTime for transition effects. Typically, this problem appears as "stuttering" transitions, or transitions with strange visual artifacts such as diagonal lines moving across the stack window.

For better results on slow machines, set the dontUseQTEffects property to true. If the dontUseQTEffects is true, Revolution uses its own built-in routines for visual effects, instead of using QuickTime. The built-in routines require less computing power and produce a good appearance on all machines.

Note: If the dontUseQTEffects property is true, visual effects generated by the answer effect command cannot be used.

When using QuickTime for visual effects, Revolution (or your application) must load QuickTime into memory before showing a visual effect. If the first visual effect is slow to appear, but subsequent effects are normal, this indicates that the system has enough power to use QuickTime visual effects, but that the time it takes to load QuickTime is slowing down the first one.

To avoid this problem, insert the following statement in a startup, preOpenStack, or openStack handler, so that it executes before any visual effects are needed:

```
get the QTEffects
```

Calling the QTEffects function loads the visual effects portion of QuickTime, so it will be available for the rest of the session. (You do not need to use the value returned by the QTEffects function; just calling it is enough to load QuickTime.)

Why is a control the wrong size when created by a handler?

See Also:

Why does an object change size?, How to create a control, clone command, create command, paste command, templateButton keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword

When Revolution creates a control, it checks the control's size. If both dimensions are less than nine pixels, Revolution resizes the new control to the default size that's set in the "Sizes and Appearance" pane of the Preferences window. This limitation ensures that when you click once in a stack window with an object tool, the object is created with the default size.

Because of the way this limitation is implemented, however, it also applies when you use the create, clone or paste command to create a new object. If the new object is smaller than 9x9 pixels at the time it is created, Revolution resizes it automatically to match the Preference setting.

For example, if you set the templateButton's height and width properties to less than 9 and then create a button, the button is resized to the default height and width. The same thing happens if you clone or paste a control whose height and width are both less than 9 pixels.

To create a control smaller than the Preferences settings, use one of the following methods:

ò Set the lockMessages to true before creating the control:

```
set the width of button 1 to 5
set the height of button 1 to 1
lock messages
clone button 1
unlock messages
```

ò Set the control's height and width after creating the control.

Important! This behavior only applies when you create a control in the Revolution development environment. If your standalone application creates a control, the minimum-size constraint does not apply.

Why is a custom property garbled in a protected stack?

See Also:

Why is a custom property garbled when switching platforms?, About custom properties and custom property sets, password property

Setting a stack's password property encrypts all the text in the stack, including any data in custom properties. If a custom property contains binary data — such as a sound file that you have stored in the custom property instead of as a separate file — that binary data will become corrupted if you password-protect the stack it's stored in.

To maintain the integrity of binary data when you want to password-protect the stack it's in, store the binary data in a separate stack. You can password-protect the stack that contains your scripts, while leaving the separate stack (containing the binary data) unprotected.

This caution does not apply to text data in custom properties, only to binary data.

Why is a custom property garbled when switching platforms?

See Also:

Why does my stack open slowly?, Why is a custom property garbled in a protected stack?, Why was a downloaded file corrupted?, About custom properties and custom property sets

Characters whose ASCII value is between 128 and 255 are not necessarily the same in the Mac OS character set and the ISO 8859-1 character set used on Unix and Windows systems. These special characters are typed with the Option key on Mac OS and OS X, and include characters used for writing languages other than English, accented characters, and some punctuation marks (such as curved quotes).

Because of this, if you create special characters in one character set and display them in the other, the appearance of the characters won't be correct because their ASCII values don't correspond. When you move a stack developed on a Mac OS or OS X system to a Windows or Unix system (or vice versa), Revolution automatically translates text in fields and scripts into the appropriate character set.

However, text in custom properties is not converted between the ISO and Macintosh character sets. This is because custom properties can contain binary data as well as text, and converting them would garble the data. If you plan to display the data in a custom property as text, you must convert it yourself when you use the stack on another platform.

To translate text that includes these characters, you use the `macToISO` and `ISOToMac` functions. The following example converts a custom property of a stack that was created on a Mac OS or OS X system so that the text can be displayed properly on a Unix or Windows system:

```
set the myCustomProp of this stack  
to MacToISO(the myCustomProp of this stack)
```


Why is Revolution taking more memory than its allocation?

See Also:

Why am I running out of memory?, Memory and Limits Reference, alwaysBuffer property, hasMemory function, heapSpace function

The Mac OS allocates a fixed amount of memory to each application. If an application requires additional memory, it can make a system call to request temporary memory for itself, over and above the application's assigned allocation. This temporary memory is included in the application's memory space, as displayed in the Finder's "About This Computer" window and in other utilities that monitor memory use.

On Mac OS systems, Revolution requests temporary memory when necessary (typically for displaying images and playing movies that require a great deal of memory, or for opening large stacks). Applications built with Revolution also use temporary memory when needed.

If for some reason you want to prevent Revolution (or your application) from requesting temporary memory, use ResEdit (or another resource editor) to edit the TMEM 128 resource in the application. Change the content of the resource from 0101 to 0001 to disable temporary memory.

Important! If temporary memory is turned off, make sure the application's memory allocation is set to at least 6M plus the total size of all the stacks that will be open at once.

Why is some text blue and underlined?

See Also:

How to change the style of text, How to create a hypertext link, How to remove the underline from linked text, Text menu > Link, link keyword, linkColor property, linkHiliteColor property, linkVisitedColor property, textStyle property, underlineLinks property

Grouped text, whose textStyle is set to `link` (either in a handler, or by choosing Text menu>Link) can be given a special appearance. By default, such text is shown colored blue and with an underline, in imitation of default settings for popular web browsers. This is because grouped text is often used to create hypertext links, like the links in a browser.

To control the underlined appearance of grouped text, use the underlineLinks property. The underlineLinks is both a global and a stack property. To remove underlining in a stack, set the stack's underlineLinks to false:

```
set the underlineLinks of stack "Text" to false
```

To remove it for all stacks that don't specify a setting, set the global property:

```
set the underlineLinks to false
```

To control the color of grouped text, use the linkColor, linkVisitedColor, and linkHiliteColor properties. Like the underlineLinks, these are global properties and also stack properties.

You can also set a stack's underlineLinks and its colors in the stack's property inspector.

Why is the card number in the window name?

See Also:

Why is there an asterisk in the window name?, View menu > Go Next, View menu > Go Prev, cantModify property, decorations property, label property, name property, number property

If a stack is modifiable (that is, if it is displayed in a normal, editable window, and its cantModify property is set to false), Revolution displays the card number in parentheses at the end of the stack's name in the window title bar, as an aid to editing. The card number does not appear if the stack has only one card.

To prevent Revolution from showing the card number, set the stack's label property. If the label property is not empty, Revolution shows the label (instead of the stack's name property) in the window title bar, without the card number.

Why is the selected text deselected?

See Also:

Why can't I select, copy, or paste text?, Why can't I leave all lines of a list field unselected?, Why is the selection lost when clicking a button?, Edit menu > Select All, Edit menu > Deselect All, autoHilite property, lookAndFeel property, platform function, select command, traversalOn property

Under certain circumstances, Revolution automatically deselects the current text selection. This can cause a problem if a handler uses the current text selection, and triggering the handler requires one of the actions that deselects the text. In this case, the handler's request for the text selection always gets an empty selection.

The selected text in a field is deselected under the following circumstances:

ò The user clicks elsewhere in a field or selects another section of text.

ò The user clicks a button whose traversalOn property is set to true. To correct this problem, set the button's traversalOn property to false. This prevents the user from pressing Tab to make the button active, but it allows the button to be clicked without losing the current text selection.

ò The user clicks a list field whose autoHilite property is set to true. To correct this problem, set the list field's autoHilite property to false. The lines of the list field will no longer highlight automatically when clicked, so you will need to insert a handler into the list field's script to highlight a clicked line:

```
on mouseDown
    set the hilitedLines of me
        to (word 2 of the clickLine)
end mouseDown
```

ò On Unix systems that are set up to use implicit focus, when the mouse pointer moves out of the stack window. Since implicit focus causes problems when operating Revolution, you should set up your system to use explicit focus when using Revolution or applications created with it.

Why is the selection lost when clicking a button?

See Also:

Why is the selected text deselected?, selectedText function, selection keyword, traversalOn property

When clicking a button while text in a field is selected, the text selection may be lost on Mac OS and OS X systems under some circumstances. This may cause a script error or other unexpected behavior if a handler in the button's script uses the selection—for example, if clicking the button boldfaces the selected text.

The problem occurs when using a standard button whose traversalOn property is set to true, with the lookAndFeel property set to `Appearance Manager`. (This problem does not occur if the lookAndFeel is `Macintosh`.)

If one of your buttons uses the selection in a mouseUp or mouseDown handler, set its traversalOn property to false to prevent this problem and allow the selection to be retained when clicking the button.

Note: Setting the button's traversalOn to false will prevent navigating to that button with the keyboard when the lookAndFeel is either `Motif` or `Windows 95`. If your application will be used on Unix or Windows systems, as well as Mac OS or OS X systems, set the button's traversalOn in an openCard or openStack handler (or use property profiles). If you set the traversalOn to false if the lookAndFeel is `Appearance Manager`, and to true otherwise, the button will operate correctly on all platforms.

Why is there a border around controls?

See Also:

Why do unwanted objects appear on new cards?, Why doesn't the Tab key move to the next field?, defaultButton property, focus command, layer property, lookAndFeel property, platform function, showFocusBorder property, traversalOn property

A control may be drawn with an extra border if it is the active control, depending on the control's showFocusBorder property and on the lookAndFeel setting. This border indicates which control has the focus.

A control becomes active when the user clicks it or presses the Tab key to move to the next control in the tab order. When a card is first displayed, the lowest control whose traversalOn property is true becomes the active control. (The lowest control is the control whose layer property is the lowest.)

For a control to become active, its traversalOn property must be set to true; a control whose traversalOn is false cannot become the active control.

ò If the lookAndFeel is set to ôWindows 95ö (Windows systems), the active control is drawn with a dashed border.

ò If the lookAndFeel property is set to ôMotifö (Unix systems), the active control is drawn with a solid border around it if its showFocusBorder is true.

ò If the lookAndFeel is set to ôMacintoshö or ôAppearance Managerö (Mac OS systems), active fields and images are drawn with a border if their showFocusBorder is true.

Note: If a button's defaultButton property is set to true, it is drawn with an additional, thick border around it. (This border is independent of the focus border described above.) The border indicates that the button will automatically be clicked if the user presses Return or Enter, and is usually used for the default button in a dialog box.

Why is there a copy of my stack file with a tilde?

See Also:

Why can't I find a stack I just saved?, Why can't I save a stack?, Why is there already a stack with the same name?, How to delete a file, File menu > Save, name property, save command

When you save changes to an existing stack file, Revolution creates a backup copy (with the same name as the file, plus a ~ character at the end of the name) before saving, then deletes the backup copy after the file has been saved. This ensures that the data will not be lost if the save operation fails.

If you find a stack file whose name ends in ~, it means the save operation wasn't completed successfully. If this happens, delete (or move) the original stack file (which may have been corrupted by the incomplete save), and rename the ~ file by deleting the ~ character from the end of the name. This replaces the original stack file with the backup copy.

Since the ~ file is a backup copy, it contains the full contents of the stack as of the last time it was saved.

Why is there a problem with line endings?

See Also:

Why do some characters change when transferred between systems?, Why is a custom property garbled when switching platforms?, CRLF constant, linefeed constant, open driver command, open file command, open process command, return constant

Different operating systems use different characters to mark the end of a line. Mac OS and OS X use a return character (ASCII 13), Unix systems use a linefeed character (ASCII 10), and Windows systems use a return followed by a linefeed. If you look at a file that doesn't have the proper end-of-line markers for your platforms, you may see lines run together, or double spacing between lines. To avoid problems when transporting a stack between platforms, Revolution ends all lines in a stack with linefeeds.

When you open a file, process, or driver, you can specify either text or binary mode. If you specify text mode, Revolution automatically uses the current system's standard end-of-line marker. This is also done when you use a file URL as a container. Revolution translates as needed between the your system's end-of-line marker and Revolution's linefeed character.

Important! Normally, you do not need to do anything special in your scripts to translate line endings: this translation is handled for you.

However, in a few situations, you may need to translate the end-of-line marker yourself:

ò If you open a file, process, or driver in binary mode and read data from it, or if you get a binfile URL, line endings are not translated for you. (This prevents accidental garbling of binary data, which may contain linefeeds and returns.)

If you need to read text data over a binary connection, and the data you're reading uses a different end-of-line marker than your current system, you will need to change the line endings appropriately in your script. One way to do this is to use the replace command:

```
replace numToChar(13) with linefeed in receivedData
```

ò Similarly, if you write data to a file, process, or driver in binary mode, or if you put data into a binfile URL, line endings are not automatically translated and you must do the translation yourself.

ò If you create a text file containing a Transcript script for use as a CGI program on a web server, you must make sure that the end-of-line marker in the file corresponds to the end-of-line marker for the server's operating system. For example, if you create the file on a Windows system for use on a Unix-based web server, you must make sure that the file's end-of-line marker is the Unix linefeed character, not the Windows CRLF.

Why is there already a stack with the same name?

See Also:

Why can't I find a stack I just saved?, Why is there already a stack with the same name?, Why is there a copy of my stack file with a tilde?, destroyStack property, mainStack property, name property

Since Revolution uses each main stack's name property to keep track of it, you cannot have more than one main stack with the same name open at one time.

If you open a main stack with the same name as a main stack already in memory, Revolution displays a dialog box asking whether you want to remove the old one from memory and open the new one. If it's not clear which stack is already open, check the following:

ò Is the stack window closed, but still in memory? (If a stack's destroyStack property is set to false, it remains in memory even after its window is closed.)

ò Did a handler read a property from the stack without opening it? (When a property is read from a stack, Revolution loads the stack into memory, although without opening a stack window for it.)

To close a stack whose window has been closed, but which remains in memory, find the stack in the Application Browser window. Control-click the stack (Mac OS or OS X) or right-click (Unix or Windows) and choose "Close and Remove from Memory" from the contextual menu to reclaim the memory that the stack is using.

Note: You can remove a stack from memory only if any other stacks in the same stack file are also closed and removed from memory.

ò Are you opening an already-open stack using a different file path? If you use a different file path to open an already-open stack, Revolution treats them as different stacks when checking to see whether the names conflict.

ò Are you opening an already-open stack after moving or renaming its file? Revolution uses the stack file's name and location in determining whether a stack is already open.

Why is there an asterisk in the window name?

See Also:

Why is the card number in the window name?, How to change a window's title, cantModify property, label property, name property

If a stack is modifiable (that is, if it is displayed in a normal, editable window, and its cantModify property is set to false), Revolution displays an asterisk (*) at the end of the stack's name in the window title bar. This asterisk indicates that the stack can be edited.

To prevent Revolution from showing the asterisk, set the stack's label property. If the label property is not empty, Revolution shows the label (instead of the stack's name property) in the window title bar, without an asterisk.

Why isn't a group listed?

See Also:

Why do unwanted objects appear on new cards?, Why doesn't a control appear?, About groups and backgrounds, Object menu > Edit Group, Object menu > Place Group, groupIDs property, groupNames property, there is a operator

Just as a group can contain controls such as buttons and fields, a group can contain other groups. If one group is part of another, the first group does not appear in the "Place Group" submenu of the Object menu, or in expressions such as the backgroundNames of this stack.

You can test whether a group is in your stack—even if it is part of another group—by using the there is a operator. For example, to check whether a group named "Foo" exists in your stack, enter the following into the message box:

```
there is a group "Foo"
```

This expression reports true if such a group exists, and false if it doesn't.

Why isn't a handler executed?

See Also:

Why can't Revolution find a handler?, Why doesn't a control get messages while the mouse is down?, Why does a mouseUp handler fail to work?, script property, set command

If a handler fails to run, it may be because the handler's message is not being sent to the object, or because the script the handler is in has not been compiled. If a handler does not run, check the following:

ò Is the message being received by the object?

To test whether the message is being received, temporarily place a statement such as the following in the first line of the handler:

```
put "Got message!" -- debugging code
```

Then perform the action that sends the message. If you see "Got message" appear in the message box, the handler is being executed. This narrows down the problem to the code in the handler itself.

ò Is the message being sent to a disabled control?

Controls whose enabled property is set to false do not receive mouse-related messages such as mouseDown, mouseUp, mouseEnter, and so on. To execute such a handler, either enable the control or send the message from another handler.

ò Is the lockMessages property set to true?

The lockMessages prevents navigation messages (such as openCard or closeStack) from being sent. Handlers for these messages are therefore not executed if the lockMessages is true.

ò Is a tool other than the Browse tool being used?

When a tool other than the Browse tool is being used, the Revolution development environment prevents many built-in messages (such as mouseEnter and mouseLeave) from being sent, so that you can edit objects without executing unwanted handlers.

ò Is the message being trapped by another object?

If another object earlier in the message path has a handler with the same name, that handler stops the message from being sent further. For example, if your stack has a mouseDown handler, but one of the cards in the stack also has a mouseDown handler, the card traps the mouseDown message before the stack has a chance to handle it. (To pass a message on to the next object in the message path, use the pass control structure.)

ò Is the script uncompiled?

When you Apply a script in the script editor, Revolution automatically compiles the script, making it ready to run. (Scripts are also compiled when you import a HyperCard stack or change the script using the set command.) If there is a compile error, the error window appears and the script is not compiled. You must fix the error and successfully compile the script before any of the handlers in the script can be executed.

Why was a downloaded file corrupted?

See Also:

Why can't I open a downloaded stack?, Why is there a problem with line endings?, binfile keyword, file keyword, ftp keyword, http keyword, libURLDownloadToFile command

If a file you download turns out to have been corrupted during the transfer—that is, if it won't open, or if trying to uncompress it results in an error message—try the following:

- ò Make sure the copy on the server is not corrupted. To double-check, download the file using another application. If the copy you download with another application is also corrupted, the file was probably damaged when it was first uploaded to the server. Notify the person who uploaded the file, or the server administrator, about the problem.

- ò Check to make sure you're using the correct URL scheme for the file. For a text file, use a file URL:

```
put URL "http://www.example.com/myfile.txt"
```

```
into URL "file:download.txt"
```

For a binary file—an application, a picture file, a word-processor file, or anything other than plain text—use a binfile URL:

```
put URL "http://www.example.com/myfile.jpg"
```

```
into URL "binfile:download.jpg"
```

Note: This same caveat applies to any operation using file and binfile URLs: you should always refer to a text file with a text URL, and to any other kind of file with a binfile URL. The reason is that different operating systems use different characters to mark the end of a line in a text file. If you use a text URL, Revolution automatically translates the end-of-line marker during the transfer. However, in binary files, changing the end-of-line marker will corrupt the file, so Revolution leaves the file data untouched when you use a binfile URL.

About the Revolution documentation

See Also:

Help menu > Documentation, Shortcut to look up Transcript term, Shortcut to open Revolution documentation

Revolution's complete documentation is provided in the form of a set of Revolution stacks and was created entirely in Revolution's development environment.

To consult the documentation, choose Help menu Revolution Documentation.

Contents:

The Documentation Window

The Section Windows

Documentation Shortcuts

Additional Resources

Legal Gibberish & Fine Print

Credits & Acknowledgements

Summary

The Documentation Window

The Documentation window, which appears when you choose Help menu Revolution Documentation, contains your roadmap to the Revolution documentation. The roadmap is a list of pages where you can find help and information. Click any of the page names to see a list of topics for each page. For information about what topics are listed on a page, hold the mouse over the page name.

To move through the pages in order, click the arrow buttons near the lower right corner of the window, or use the arrow keys. To go back to the list of pages, click the "Roadmap" button next to the arrows.

If you're brand-new to Revolution, start with "Welcome to Revolution". This page contains a list of recommended readings and tutorials to get you up and running with Revolution development, depending on your past experience. It appears automatically the first time you open the documentation.

To find a topic by category, go to "Documentation by Category". This page lists every topic in the documentation under categories like printing, menus, databases, and more.

If you're looking for a language term, go to "Transcript Language Dictionary". You can use this page to search for terms and to scan parts of the language such as commands, functions, properties of each object, operators, and so on.

Tip: You may find that you use a particular page of the documentation—such as "Transcript Language Dictionary"—more often than the rest. To make that page appear immediately whenever you open the Documentation window, check the box at the bottom of the window labeled "Show this page when documentation opens". The next time you open the Documentation window, you'll go directly to that page without seeing the list of pages. (Click the "Roadmap" button to return to the list of pages.)

The Section Windows

RevolutionÆs documentation includes thousands of topics. Each topic belongs to one of the following sections:

Encyclopedia: Detailed background information on specific topics
Transcript Dictionary: Complete reference to the Transcript language
Transcript Cookbook: Annotated examples of Transcript code
How To: Step-by-step instructions for common tasks
Troubleshooting: Solutions for common problems
Tutorials: Introduction to Revolution development and scripting
Shortcuts Reference: Handy keyboard shortcuts
Menu Reference: For each menu item in the development environment
Glossary: Definitions of technical terms used in the documentation
Tips: Useful techniques and pointers

Each of these sections is presented in a separate window, so you can easily move back and forth between sections.

You can find a listing of all topics in all sections on the [Development Guide](#) page of the Documentation window. A few of the sections also have their own page in the Documentation window.

Each documentation window (including this one) has a control bar at the top. Using the control bar, you can search for a topic in the current section, return to previously-visited topics in any section, or look up related topics.

Tip: The [See Also](#) menu at the top of each documentation window includes topics in all sections of the documentation. If youÆre not sure what youÆre looking for, but you do know a related topic, check that topicÆs See Also list. For example, if you want to find the Transcript term that performs an action equivalent to the Save menu item, look up the menu item in the Menu Reference and check its See Also list for Transcript terms.

Looking up terms in the documentation

Transcript language terms are boldfaced throughout all sections of the documentation. Click a bold term to look up the term in the Transcript Dictionary.

When you move the mouse pointer over a word thatÆs in the glossary, the word is highlighted and the cursor changes to a pointing hand. Clicking the word displays the glossary entry. (To close the glossary, click again.)

Menu items are boldfaced. Click a menu itemÆs name to look it up in the Menu Reference.

The documentation contains references to Internet standards and other documents that are available on the Web. The URLs of such documents are blue and underlined. To open the document in your default browser, click its URL.

Tip: To copy text from the documentation, just select the text you want to copy and choose Edit menu Copy. You canÆt change the text, but you can select it.

The control bar at the top of each documentation window contains a Find box. To locate a topic title in that section, enter a term in the Find box and press Return. To search the full text of the section, hold down the Option key (Mac OS and OS X) or Alt key (Unix and Windows).

Notes and tips

Throughout the documentation, you'll see notes that give you special information. These notes are indicated as follows:

Note: Provides some additional information that may be useful, or emphasizes a key point.

Tip: Tells you about an alternative way to do something or describes a shortcut.

Important! Describes an aspect of Revolution that may confuse you if you don't know about it, and tells you how to avoid unexpected behavior.

Caution! Warns you about a potential problem that might cause a script error or an unexpected loss of data.

Transcript Dictionary special features

The Transcript Dictionary has some additional features:

- ò The icons near the top of the dictionary window for Mac OS, OS X, Unix, and Windows indicate which platforms each Transcript term is supported on.

- ò The syntax for each language term (other than keywords) is provided in boldface typewriter text. The syntax description uses the standard conventions to indicate parts of the syntax:

- [] Square brackets enclose optional portions.

- { } Curly braces enclose sets of alternatives from which you choose.

- | Vertical bars separate different alternatives.

Italics indicate placeholders that you don't enter literally.

- ò The icon at the upper right corner of the Transcript Dictionary window switches between full view and quick-reference view. The quick-reference view is a small palette that displays the term's syntax and a brief description. You can keep the Dictionary open in quick-reference view without blocking the rest of the screen.

Clicking the icon again switches back to full view so you can check examples, platform support, and information about parameters and return values.

Tip: When you open the Dictionary from within the script editor, it automatically opens in quick-reference view.

Documentation Shortcuts

To open the documentation:

Press Command-? (on Mac OS or OS X systems).

Press Control-? (on Unix and Windows systems).

Press the F1 key.
Press the Help key.

To scan through the topics in the current section:
Press the left and right arrow keys.

To scroll up and down (in topics that have scrollbars):
Press the up and down arrow keys.

To copy a selection of text:
Select the text and choose Edit menu Copy.

To search for a topic in the current section:
Type a word or phrase in the Find box at the top of the window and press Return or Enter. Continue pressing Return or Enter to search for more topics whose title contains the word. To search the full text, hold down the Option key (Mac OS or OS X) or Alt key (Unix or Windows).

To return to a previously-visited topic:
Choose a topic from the Go Back menu at the top of the window.

To look up related topics:
Choose a topic from the See Also menu at the top of the window.

To look up a Transcript term in the script editor:
Command-click the word (on Mac OS or OS X systems).
Control-click the word (on Unix or Windows systems).

Additional Resources

If you have a question that isn't answered in this documentation, you can find additional information to help you in developing your applications. You can also connect with the Revolution developer community, which includes many experienced developers who offer their expertise for the common good to answer questions and help their fellow developers.

Your starting point is Revolution's web site, located at [<http://www.runrev.com>](http://www.runrev.com). Here you'll find example code, information on planned enhancements, and the latest updates.

Developer community

Mailing lists are available for getting help from other developers, for discussing feature requests and future directions, and for receiving announcements from Runtime Revolution. To join the Revolution mailing lists, visit [<http://www.runrev.com/revolution/developers/maillinglists/>](http://www.runrev.com/revolution/developers/maillinglists/).

Tip: You can use the Google search engine to search the list archives, using [<http://www.google.com/advanced_search?q=site:lists.runrev.com>](http://www.google.com/advanced_search?q=site:lists.runrev.com).

Keeping current

To make sure you always have the latest documentation, choose Help menu Check for Updates Online. If a new release of Revolution or the documentation is available, Revolution displays a message telling you how to download it.

Technical support

Runtime Revolution offers on-line and on-site technical support to all users to assist you in getting the development environment working on your system.

The Enterprise Edition includes additional technical support for more advanced problems and questions. And with any edition, you can purchase a separate technical support contract for advanced support.

To request support, choose Help menu Revolution Support, or send email to support@runrev.com.

Tip: Si vous faites partie de la communauté Francophone, et préférez obtenir de l'aide ou support en Français, envoyez un email à Frédéric Rinaldi frederic@runrev.com.

Legal Gibberish & Fine Print

Version 2.1

Documentation © 2000-2003 Runtime Revolution Ltd. All rights reserved.

This documentation may be altered or reproduced only for the personal use of the license holder, and may not be redistributed in any form without the explicit written permission of the copyright holder.

Apple, HyperCard, Macintosh, QuickTime, Mac OS, OS X, and the Mac OS and OS X logos are trademarks of Apple Computer, Inc. PostScript is a trademark of Adobe Systems Incorporated. Motif and UNIX are registered trademarks of The Open Group. Windows and the Windows logo are trademarks of Microsoft Corporation. All other trademarks are the property of their respective owners.

Credits & Acknowledgements

Documentation by Jeanne A. E. DeVoto <jeanne@runrev.com>

Transcript Cookbook by Jeanne A. E. DeVoto and Jacqueline Landman Gay

Tutorials by Richard Gaskin

Thanks to...

Scott Raney, Kevin Miller, Dave Cragg, Andu Novac, Tuviah Snyder, and Geoff Canyon for expert technical assistance

Matt Denton for designing the print version of the documentation
(any problems with the print version are due to my implementation, not to Matt's excellent design)

Heather Williams and Steve Wagenseller for technical review of the Tutorials

Rob Cozens for technical review of the Transcript Dictionary

The Revolution List for extensive commentary and testing

The MetaCard List for tips and information

Summary

In this topic, you have learned that:

• You access the documentation through the main Documentation window, which is shown when you choose Help menu Revolution Documentation.

ò Each section of the documentation is presented in a separate window.

ò You can use the control bar at the top of each documentation window to search for a topic in that section, back up to the previous topic, or look up related topics.

ò To find more help, start with the Revolution web site at <<http://www.runrev.com>>.

About Revolution system requirements

See Also:

About Revolution applications under Windows 3.1, Memory and Limits Reference, Supported Platforms Reference, How to estimate how much memory your application will need, Why am I running out of memory?, Why is Revolution taking more memory than its allocation?, destroyStack property, diskSpace function, files function, hasMemory function, heapSpace function

This topic describes the system requirements and recommendations for the Revolution development environment, and for applications created in Revolution.

The information on development environment requirements is the same as the information contained in the Read Me First text file that ships with Revolution. It is provided here for convenient reference.

Contents:

Development Environment Requirements
Requirements for Applications

Development Environment Requirements

All operating systems require:

- ò 800x600 or larger monitor
- ò 8-bit (256-color) or higher

Note: Memory and disk requirements below are for the Revolution development environment, not for applications created by Revolution.

Requirements for Mac OS systems

You can develop on any Mac OS system that includes:

- ò Any 68K or PPC Macintosh or clone
- ò Operating system: Mac OS 7.1 or later
- ò Memory:
 - at least 48M total (128M total recommended)
 - at least 20M free (48M free recommended)
- ò Disk space: 20M free
- ò QuickTime 3.0 or later required for video features (4.1 recommended)

Requirements for OS X systems

You can develop on any OS X system that includes:

- ò Any OS X-capable Macintosh
- ò Operating system: OS X 10.0.3 or later
- ò Memory: 128M total
- ò Disk space: 20M free
- ò QuickTime required for video features

Requirements for Unix & Linux systems

You can develop on any Unix system that includes:

- ò One of the following operating systems:

- FreeBSD or BSDI

- HP-UX 10.20 or later

- SGI IRIX 5.3 or later

- Linux Intel 1.2.13 ELF or later

- AIX 3.2.3 or later

- Solaris x86 2.5 or later

- SPARC Solaris 2.3 or later

- SPARC SunOS 4.1.x or later

- ò Memory: 48M total

- ò Disk space: 20M free

- ò Xanim required for video features

Requirements for Windows systems

You can develop on any Windows system that includes:

- ò Operating system: Windows 95, 98, Me, NT, 2000, or XP

- ò Memory: 48M total

- ò Disk space: 20M free

- ò QuickTime 3.0 or later required for video features (4.1 recommended)

Requirements for Applications

The standalone applications you create with Revolution are true double-clickable executables and do not require a runtime library, VM, ôplayerö application, DLLs, extensions, or other extra files to be installed.

Operating systems

Revolution applications can run on all the operating systems and versions listed above:

- ò Mac OS 7.1 or later

- ò OS X (any version)

- ò FreeBSD or BSDI

- ò HP-UX 10.20 or later

- ò SGI IRIX 5.3 or later

- ò Linux Intel 1.2.13 ELF or later

- ò AIX 3.2.3 or later

- ò Solaris x86 2.5 or later

- ò SPARC Solaris 2.3 or later

- ò SPARC SunOS 4.1.x or later

- ò Windows 95, 98, ME, NT, 2000, or XP

Revolution applications can also run, with some limitations, under Windows 3.1. (See the topic ôAbout Revolution applications under Windows 3.1ö for details.)

Memory use

The memory required by a Revolution application will vary, depending on the nature of your application. The total depends on many factors, including the platform, the complexity of your code, which Revolution custom libraries you include, how many windows you have open at once, and your use of media such as images and players. Most Revolution applications do not require as much memory as the development environment.

As a very rough rule of thumb, add up the size of the stacks to be loaded into memory (and the picture and movie files to be displayed in referenced controls) at any one time, and add 4M to obtain a loose approximation:

	total size of loaded stacks
+	total size of external files displayed
+	4 megabytes
=	minimum memory required (very approximate)

Important! The formula above is not definitive and is intended only as a rough guideline. The only way to determine exactly how much memory your application requires is to test the finished application in real-world conditions.

Disk space use

The minimum disk space required by a Revolution application depends on which platform you build it for, and on which options you choose on the Inclusions tab in Step 3 of the Distribution Builder. If you do not select any resources to include, the minimum disk space varies between approximately 1.5M (for a Windows application) and 3.3M (for an IRIX application).

As a rule of thumb, add the size of all stack files to be built into your standalone (and, for Mac OS and OS X standalones, the size of any resources you add) to the following minimums to obtain an estimate of required disk space:

Platform:	Minimum standalone size:
Mac OS PPC	2M
Mac OS 68K	2M
Mac OS �fat�	2M
OS X	2.1M
BSD	1.6M
HP-UX	2.7M
IRIX	3.3M
Linux Intel	2M
IBM RS-6000	2.7M
SPARC SunOS	1.9M
SPARC Solaris	1.9M
Solaris (Intel)	2.1M
Windows	1.5M

About Revolution applications under Windows 3.1

See Also:

About Revolution system requirements

Revolution-built applications can run under Windows 3.1, with some limitations. This topic describes limitations and other information about using Revolution applications under Windows 3.1.

Important! The limitations described in this topic apply only to version 3.1 of Windows. Later Windows versions (from Windows 95 on) are fully supported.

Contents:

Requirements for Windows 3.1

Unsupported Features

Known Windows 3.1 Issues

Summary

Requirements for Windows 3.1

The following requirements apply to using Revolution applications on Windows 3.1 systems:

- ò MicrosoftÆs Win32s package must be installed.
- ò The look and feel is Windows 95, not native Windows 3.1.
- ò A TCP/IP package must be installed (WINSOCK.DLL).

If no WINSOCK.DLL is installed, a warning appears during startup. However, since Revolution applications do not use the functions in this DLL (unless you attempt to use http or ftp URLs), you can suppress the warning by simply making a copy of another DLL and naming it WINSOCK.DLL.

Important! RevolutionÆs Windows 3.1 support is primarily intended to allow deployment of Revolution applications on Windows 3.1 systems. Developing applications under Windows 3.1 is not recommended.

Unsupported Features

The following features of Revolution are not supported by applications under Windows 3.1:

- ò All printing-related commands, functions, and properties
- ò The ôdissolveö, ôstretchö, and ôshrinkö visual effects
- ò All QuickTime visual effects
- ò The shell function
- ò The open process command
- ò The load command
- ò The post command
- ò The http and ftp URL schemes
- ò The import snapshot command

Known Windows 3.1 Issues

Known problems when using a Revolution application under Windows 3.1 include:

- ò Some window navigation features may not work correctly due to incompatibilities in Win32s.
- ò Windows don't always come to the front when you click them. (To partially work around this problem, set the raisePalettes property to false.)
- ò Pressing Alt-Tab doesn't display the Task Manager.

Summary

In this topic, you have learned that:

- ò You can use Revolution-based applications under Windows 3.1, with some limitations on features.
- ò To use Revolution applications on a Windows 3.1 system, you must first installed Win32s and WINSOCK.DLL.

About installing Revolution

See Also:

About upgrading Revolution, Supported Platforms Reference

This topic describes the procedure to install the Revolution development environment.

Note: Obviously, if you are reading this, you have already installed Revolution. The information in this topic is the same as the information contained in the "Read Me First" text file that ships with Revolution. It is provided here for convenient reference.

Contents:

Overview

Installation on Mac OS Systems

Installation on OS X Systems

Installation on Linux Systems

Installation on Unix Systems

Installation on Windows Systems

License Information

Overview

To install Revolution, you download the build for your platform from the Runtime Revolution web site at <http://www.runrev.com>, then decompress it (for Mac OS or Unix systems), copy the contents (for OS X systems), or run the Revolution installer (for Windows systems).

When you first run Revolution, you can configure it as the 30-day Evaluation Edition. The Evaluation Edition allows unlimited access to Revolution's features for a 30-day trial period. After 30 days, you must purchase a license to continue using Revolution.

Installation on Mac OS Systems

To install Revolution on your system, download the file

<http://www.runrev.com/revolution/downloads/distributions/2.1.2/revolution.sit> from the Runtime Revolution web site, then unstuff the file with StuffIt Expander. (You can also download Revolution from the Runtime Revolution FTP site at [ftp.runrev.com](ftp://ftp.runrev.com).) This creates a folder called "Revolution", which contains all components of the development environment. You can place this folder anywhere on your disk.

Important! Do not unstuff the file on a Windows system. Unstuffing on a Windows system will not preserve the resource fork, even if you then move the unstuffed folder to a Mac OS system. If you are installing Revolution on a Mac OS or OS X system, you must unstuff the file on a Mac OS or OS X system.

Start up the Revolution development environment by double-clicking the Revolution application inside the folder. If you want to uninstall Revolution, simply drag the Revolution folder to the Trash.

Note: The first time Revolution starts up, it sets a preference in the File Exchange control panel to associate Revolution files created on other platforms with the application. (Specifically, it maps the

extension `.rev` to the creator signature `Revo` and the type signature `RSTK`.) This ensures that downloaded Revolution files, and files you copy from PC-formatted media, will be properly registered in the Finder as belonging to Revolution.

Installation on OS X Systems

To install Revolution on your system, download the file
<<http://www.runrev.com/revolution/downloads/distributions/2.1.2/revolutionosx.dmg>>
from the Runtime Revolution web site, then double-click the file. (You can also download Revolution from the Runtime Revolution FTP site at <ftp.runrev.com>.) This mounts a disk image on your desktop. To install Revolution, open the disk image and copy the contents to your Applications folder.

Start up the Revolution development environment by double-clicking the Revolution application. If you want to uninstall Revolution, simply drag the Revolution folder to the Trash.

Installing for Darwin only

To install Revolution on a Darwin system, download the file
<<http://www.runrev.com/revolution/downloads/engines/2.1.2/darwin.gz>> and decompress it into a directory.

The Darwin version of Revolution is intended for development of command-line scripts, and does not support the Revolution development environment or user interface.

Installation on Linux Systems

To install Revolution on your system, download the file
<<http://www.runrev.com/revolution/downloads/distributions/2.1.2/linux.tgz>> and gunzip and untar it. (You can also download this file from the Runtime Revolution FTP site at <ftp.runrev.com>.)

To start up the Revolution development environment, run `revolution`. If you want to uninstall Revolution, simply delete `Revolution/Es` directory.

RPM Linux installation

There are two options for installation on Intel-based Linux systems. You can use the Redhat Package Manager (RPM) distribution or the standard `.tgz` distribution described above.

To install the RPM distribution, download the file
<<http://www.runrev.com/revolution/downloads/distributions/2.1.2/revolution-2.1.2-1.i386.rpm>>
from the Runtime Revolution web site.

Then install Revolution as root by running the following shell command:

```
rpm -ivh revolution-2.1.2-1.i386.rpm
```

Revolution is installed in the `/opt` directory.

To start up the Revolution development environment, run `revolution`. If you want to uninstall Revolution, run the following shell command:

```
rpm -e revolution-2.1.2-1
```

Installation on Unix Systems

To install Revolution on your system, first download the file
<<http://www.runrev.com/revolution/downloads/distributions/2.1.2/unix.tgz>> and gunzip and untar it.

Next, download the correct file for your Unix flavor from the table below, then gunzip it:

FreeBSD and BSDI:

<<http://www.runrev.com/revolution/downloads/engines/2.1.2/Bsd.tgz>>

HP-UX:

<<http://www.runrev.com/revolution/downloads/engines/2.1.2/Hp9k700.gz>>

SGI IRIX:

<<http://www.runrev.com/revolution/downloads/engines/2.1.2/Iris.gz>>

AIX:

<<http://www.runrev.com/revolution/downloads/engines/2.1.2/rs6000.gz>>

Solaris x86:

<<http://www.runrev.com/revolution/downloads/engines/2.1.2/SolIntel.gz>>

SPARC Solaris:

<<http://www.runrev.com/revolution/downloads/engines/2.1.2/SolSparc.gz>>

SPARC SunOS:

<<http://www.runrev.com/revolution/downloads/engines/2.1.2/SPARC.gz>>

Note: You can also download these files from the Runtime Revolution FTP site at <ftp.runrev.com>.

After decompressing the correct file for your flavor of Unix, rename it `örevolutionö`, and use it to replace the `örevolutionö` file in the folder you previously untarred.

To start up the Revolution development environment, run `örevolutionö`. If you want to uninstall Revolution, simply delete `RevolutionÆs` directory.

Installation on Windows Systems

To install Revolution on your system, download the file

<<http://www.runrev.com/revolution/downloads/distributions/2.1.2/revsetup.exe>>

from the Runtime Revolution web site, then double-click the file to install it. (You can also download Revolution from the Runtime Revolution FTP site at <ftp.runrev.com>.) This creates a folder called `öRevolutionö`, which contains all components of the development environment.

Important! On Windows NT systems, Service Pack 6 or later is required for the standard Revolution installer. If you are running an earlier version of Windows NT, download the file
<<http://www.runrev.com/revolution/downloads/distributions/2.1.2/revolution.zip>>,
then unzip the file.

Start up the Revolution development environment by double-clicking the Revolution application inside the folder, or choose `öPrograms > Revolution > Revolutionö` from the Start menu.

If you want to uninstall Revolution, remove it using the Add/Remove Programs Control Panel.

License Information

When you start up Revolution for the first time, you are given the option to configure your copy as an Evaluation Edition for thirty days.

The Evaluation Edition

When you start up Revolution for the first time, you are given the option to configure your copy as an Evaluation Edition for thirty days. The Evaluation Edition operates with the full features of Revolution Enterprise during the thirty-day evaluation period.

After the thirty-day period expires, the Evaluation Edition stops running. To continue to use Revolution after the evaluation period has passed, you must purchase a license.

Note: Applications you create with the Evaluation Edition may not be commercially distributed.

Licensing Revolution

To use Revolution beyond the thirty-day evaluation period, you must purchase a license for Revolution. Several editions—Revolution Express, Revolution Studio, and Revolution Enterprise—are available, as well as multi-user, cross-grade, and educational discounts. To find out more about the available options, or to purchase a license, choose Help menu License Revolution and follow the instructions in the dialog box.

If you are upgrading Revolution from a previous version, choose Help menu License Revolution to re-enter your license key.

About upgrading Revolution from a previous version

See Also:

About installing Revolution, About what's new in version 2.0-2.0.2, Help menu > License Revolution..., Help menu > Check for Updates

This topic describes the update policy to version 2.1 from previous versions, and how to upgrade your copy of Revolution.

Tip: To upgrade from the Evaluation Edition to a paid license, choose Help menu License Revolution and click "Purchase a License" for information about your options.

Contents:

Updating Your Current License

Upgrading to the Studio or Enterprise Editions

How to Update to Version 2.1

Updating Your Current License

To purchase an update to the current version, choose Help menu License Revolution.

Note: If you do not update your license, your copy of a previous version of Revolution (and any applications you develop with it) will continue to work, but you will not be able to use new features or take advantage of other improvements in version 2.1.

Revolution Enterprise

Revolution Studio

If you purchased a license for Revolution Enterprise or Studio 2.0.2, the update to version 2.1 is free for you.

Revolution Express

If you purchased a license for Revolution Express 2.0.2, the cost of the 2.1 update is \$19.95. To purchase the update, choose Help menu License Revolution and click "Purchase a License".

Older editions

If you purchased Revolution 2.0.1 or earlier and have the Professional Edition, Small Business Edition, Student/Teacher Edition, Professional Educational Edition, or Classroom K-12 Edition, a new license key is required for you to use version 2.0.2 and later.

If you purchased a Revolution license (for version 2.0.1 or earlier) less than a year before the August 2003 release of version 2.1, the update to Revolution 2.1 is free for you. The license key that was sent to you for version 2.0.2 will also unlock version 2.1. (If you did not receive a license key for version 2.0.2, please contact support@runrev.com for a key.)

Professional Edition:

If you purchased a Professional Edition license less than a year before the August 2003 release of version 2.1, you are entitled to a free update to Revolution Enterprise.

If your Professional Edition license is expiring, you are eligible to update to a Revolution Enterprise license for an update fee of \$299. This includes all updates for the following year. (If you purchase an update before your license expires, the year begins on the original expiration date.)

Professional Educational Edition:

If you purchased a Professional Educational Edition license less than a year before the August 2003 release of version 2.1, you are entitled to a free update to the current version of Revolution Enterprise.

If your Professional Educational Edition license is expiring, you are eligible to update to a Revolution Enterprise license for an update fee of \$149. This includes all updates for the following year. (The restrictions on commercial use still apply.) If you purchase an update before your license expires, the year begins on the original expiration date.

Small Business Edition:

If you purchased a Small Business Edition license less than a year before the August 2003 release of version 2.1, you are entitled to a free update to the current version of Revolution Studio, with the added ability to develop on any supported platform.

If your Small Business Edition license is expiring, you are eligible to update to the current version of Revolution Studio by paying a \$299 update fee. This fee includes all updates for the following year. (If you purchase an update before your license expires, the year begins on the original expiration date.) If you choose this option, you will receive a license key that allows you to use your copy of Revolution Studio on any supported platform.

Note: This option is available only to current Small Business Edition users. If you purchase a new license for Revolution Studio, you cannot purchase updates for a year at a time, and your license key is specific to the platform you purchased Revolution for.

Classroom K-12 Edition:

If you purchased a Classroom K-12 Edition license less than a year before the August 2003 release of version 2.1, you are entitled to a free update to the current version of Revolution Studio, with the added ability to develop on any supported platform. (The restrictions on commercial use still apply.)

Student/Teacher Edition:

If you purchased a Student/Teacher Edition license less than a year before the August 2003 release of version 2.1, you are entitled to a free update to the current version of Revolution Studio, with the added ability to develop on any supported platform. (The restrictions on commercial use still apply.)

Upgrading To Another Edition

You can upgrade from any current edition of Revolution to a more capable version at any time by paying the difference in price.

Upgrades from one edition to another

If you have Revolution Express, you can upgrade to the Studio or Enterprise edition by paying the difference in price.

If you have Revolution Studio, you can upgrade to the Enterprise edition by paying the difference in price.

Note: If you are entitled to a multi-user or education discount, or both, use the discounted price as the upgrade price. For example, if you purchased Revolution Studio with an education discount, you may upgrade to Revolution Enterprise by paying the difference between your initial payment for Studio and the education price for Enterprise.

Upgrades from educational to commercial licenses

Copies of Revolution purchased with an education discount are restricted to noncommercial use only. You may not develop commercial software under an education license.

To remove this restriction, you must upgrade your education license to a regular, full-price license. If you have any edition of Revolution with an educational discount, you can upgrade to a full commercial license with no restrictions on use by paying the difference in price (that is, the amount of the original discount).

MetaCard cross-grades

If you are a current MetaCard customer, the next time you purchase a yearly update for your MetaCard subscription, you will also receive a license key for the current version of Revolution Enterprise. This lets you cross-grade to Revolution Enterprise from MetaCard at no charge.

How to Update to Version 2.1

To update from a previous version, follow these steps:

1. Install 2.1:

Download and install Revolution 2.1 according to the instructions in the Read Me file.

2. Copy custom libraries:

If you have created libraries for the Image Library or Object Library, you will need to move them to the Revolution 2.1 folder. These libraries are stored in the folders `usericonsö` and `userobjectsö`, which are in the folder `revolution/components/save/ö`. Copy these folders from your old Revolution folder to your new Revolution 2.1 folder.

After doing this, you can delete the old Revolution folder if you wish.

3. Start up Revolution.

4. Enter your license key:

When you start up Revolution, a dialog box asks whether you want to configure your copy as an Evaluation Edition or provide a license key. Choose the option to provide a key, and enter your name, organization (if applicable), and license key to unlock your copy of Revolution.

If you have not received a key, have misplaced your key, or encounter any problems, contact support@runrev.com for help.

About what's new in version 2.1-2.1.2

See Also:

About what was new in version 2.0-2.0.3, About what was new in version 1.1.1, About what was new in version 1.1, How to report a bug or request a feature, How to request technical support

This topic describes changes to Revolution between version 2.0.2 (July 2003) and 2.1.2 (November 2003).

The information in this topic is the same as the information contained in the `Read Me First` and `What's New` files that shipped with version 2.1.2.

Contents:

- Overview of Version 2.1.2

- Overview of Version 2.1.1

- Overview of Version 2.1

- New Features

- Issues Addressed in This Version

- Limitations and Known Issues

Overview of Version 2.1.2

Version 2.1.2 is primarily a bug fix release. New features include:

- ò The `filter` command can now be used to get all lines that don't match a wildcard expression.

- ò The `uniDecode` and `uniEncode` functions can now be used to encode and decode Polish text.

- ò The `revMacFromUnixPath` and `revUnixFromMacPath` functions now have an optional parameter that specifies, on OS X systems, whether to use OS X disk conventions or classic Mac OS conventions.

The Externals SDK is no longer included with the Revolution installation. You can download this SDK from the Runtime Revolution website at

<<http://www.runrev.com/revolution/downloads/distributions/sdk/>>. (More documentation on creating externals will be forthcoming in a future version.)

The engine version for Revolution 2.1.2 is 2.5.1 build 3.

New Transcript terms

None.

Changed and extended Transcript terms

`filter` command, `revMacFromUnixPath` function, `revUnixFromMacPath` function, `uniDecode` function, `uniDecode` function

Removed Transcript terms

None.

Issues addressed in this version

For up-to-date information on bug fixes, consult the bug reporting system at

<<http://www.runrev.com/revolution/developers/bugdatabase/>>. A list of bug fixes for this version will also be available shortly on the Runtime Revolution website.

ò Compatibility problems with Mac OS versions older than 8.5, and OS X versions older than 10.2, have been addressed.

Overview of Version 2.1.1

Version 2.1.1 was a testing-only release, which was renumbered 2.1.2 for final release. (This was done because during testing, a beta release numbered 2.1.1 final was inadvertently available for download for a few hours. The final release was therefore renumbered, to avoid any ambiguity for users who downloaded this beta release.)

Overview of Version 2.1

Version 2.1 is a feature release. New features include:

- ò OS X drawers
- ò Several new window options
- ò OS X applications follow Aqua guidelines more closely

The edition lineup and features have changed as of version 2.0.2. For details, see the ôLicense Agreementö file in the Revolution folder, or visit the Runtime Revolution website at <<http://www.runrev.com>>.

Important! Whenever you upgrade to a new version, be sure to make a backup copy of all your stack files before saving them with the new version. This ensures that in case of a problem or incompatibility, you can go back to an older version temporarily until the problem is resolved.

The format of stack files has not changed between version 2.0 and 2.1. This means that you can open and use version 2.1 stack files in version 2.0, although features that are new for 2.1 will not work in earlier versions.

The engine version for Revolution 2.1 was 2.5.1 build 1.

New Transcript terms

drawer command, export snapshot command, fontLanguage function, hide taskbar command, liveResizing property, metal property, minimizeBox property, show taskbar command, systemWindow property

Changed and extended Transcript terms

closeBox property, decorations property, icon property, pass control structure, revOpenDatabase function, shadow property, sheet command, zoomBox property

Removed Transcript terms

None.

New Features

This section describes the new features introduced in version 2.1 in detail.

New window options

ò On OS X systems, you can set a stack's metal property to true to display it in a textured window. (See Apple's Aqua User Interface Guidelines for more information about when to use the metal style.)

ò On OS X, Windows, and Unix systems, you can set a stack's systemWindow property to true to make it float above all other applications. A system window always appears in front of other windows, even when its application is not in the foreground.

ò On OS X systems, you can set a stack's shadow property to true to eliminate the normal window drop shadow.

ò The decorations property now supports options to support these three properties.

ò Several new properties have been added to allow direct control of the close box, zoom box/maximize box, and collapse box/minimize box. The closeBox and zoomBox properties are now fully implemented, and the minimizeBox property has also been introduced. (You can also set these options using the decorations property of a stack.)

Drawers

Drawers—subwindows that slide out from an edge of a window—are now supported on OS X systems. A drawer is implemented as a separate stack, and is shown using the new drawer command.

(On Mac OS, Unix, and Windows systems, the drawer command displays the stack as a separate window.)

To slide the drawer back in, close the stack.

Important! The syntax of the drawer command has changed since 2.1b3. See the Transcript Dictionary for a description of the new syntax.

Aqua guidelines compliance

ò OS X applications now automatically delete the last two lines in the second menu of a stack's menu bar, if the last line starts with "Preferences". This is to prevent duplication of the "Preferences" menu item in the OS X Application menu. (Normally, the second menu is the Edit menu, and the last two menu items are a separator line and "Preferences".) This lets you use the same menu bar for all platforms without breaking user-interface guidelines for OS X, which call for the "Preferences" menu item to be in the Applications menu and not the File menu.

Tip: If your application's user interface is presented in a language other than English, set the name of the Edit menu button to "Edit", and set its label to the correct translation. This ensures that the engine can find the Edit menu, while making sure that the menu is shown in the correct language.

ò Changing the state of the minimize button in a window's title bar is now supported on OS X systems. If the decorations property is set to exclude the minimize button, or the minimize property is set to false, the minimize button is dimmed (rather than removed).

ò The icon property is now supported as a stack property on OS X systems. The image specified by a stack's icon appears in the dock when the stack window is minimized.

ò The icon property can also now be used as a global property on OS X systems. The image specified by the icon is used as the application's dock icon.

Documentation changes

ò The by-category listing of all documentation topics in the main documentation window used to be called "Development Guide", and can now be found under "All Documentation by Category".

ò The Documentation Search window can now be accessed from the main documentation window, instead of the Plugins menu.

ò You can now access the complete listing for a topic's category in the topic's See Also menu.

ò You can now hide or show all documentation windows with a single keystroke. To hide all open documentation windows, hold down the Option key (on Mac OS or OS X systems) or Alt key (on Unix or Windows systems) while using any of the key combinations that normally open the documentation:

Command-Option-? or Control-Alt-?

Option-F1 or Alt-F1

Option-Help or Alt-Help

You can also hold down the Option or Alt key while choosing Help menu Documentation, or while clicking "Documentation" in the toolbar.

Miscellaneous Transcript changes and additions

ò The new fontLanguage function returns the language that a font displays:

put the fontLanguage of "Osaka" -- returns "Japanese"

ò The pass control structure now supports passing a message directly to the engine without going through the rest of the message path, using the form pass...to top.

ò You can now specify a parent stack with the sheet command, to make a sheet appear in a stack other than the defaultStack.

Miscellaneous other changes and additions

ò All editions of Revolution can now use MySQL, PostgreSQL, and Valentina databases with no limitations, as well as accessing any database via ODBC. Direct access to Oracle databases is limited to Revolution Enterprise.

Note: The Free Edition is no longer available.

Issues Addressed in This Version

For a comprehensive list of bug fixes, consult the bug reporting system at <http://www.runrev.com/revolution/developers/bugdatabase/>.

Compatibility issues

ò For OS X builds, if you specify "Move substacks into individual files" in Step 2 of the Distribution Builder, the files are placed inside the application bundle rather than in a separate folder.

Development environment issues

ò The Externals SDK is no longer included with the Revolution installation. You can download this SDK from the Runtime Revolution website at [<http://www.runrev.com/revolution/downloads/distributions/sdk/>](http://www.runrev.com/revolution/downloads/distributions/sdk/). (More documentation on creating externals will be forthcoming in a future version.)

Other issues

ò Various Unicode issues have been addressed.

ò On Mac OS and OS X systems, pulldown menus in a window are now drawn by operating-system routines, as long as the menu button's `showBorder` property is set to true and its `borderWidth` is not zero. (Pulldown menus in the menu bar are always drawn by the operating system.)

Limitations and Known Issues

Please report other bugs or issues through the bug reporting system at [<http://www.runrev.com/Revolution1/bugzilla/>](http://www.runrev.com/Revolution1/bugzilla/). Also check the bug reporting system for a list of current open bugs and feature requests.

For interim updates, check the updates directory at [<http://www.runrev.com/revolution/downloads/distributions/2.1/updates/>](http://www.runrev.com/revolution/downloads/distributions/2.1/updates/), or choose Help menu>Check for Updates.

Issue: Use of the shell function on Windows systems with Norton AutoProtect installed may cause a crash.

Workaround: If you need to use the shell function on Windows systems, disable Norton AutoProtect.

Issue: On Mac OS systems, the InputSprocket extensions have been reported to cause instability with some applications, including Revolution.

Workaround: If you experience instability and you do not require the sprocket extensions, try turning these extensions off in the Extensions Manager control panel.

Issue: Some Windows graphics drivers have compatibility problems with Revolution, in particular with images. The primary symptom is that images are drawn in the wrong colors.

Workaround: If you see this symptom, make sure you have the latest drivers from the vendor of your graphics card, and try turning off graphics acceleration (in the Performance tab in the System Control Panel). If this does not fix the problem, please report the bug to your graphics card vendor.

Issue: Some Windows versions show font problems when printing text. In particular, Windows XP printouts may have blank lines between lines of text or half-length printed lines. Printed text on Windows ME or Windows 98 systems may be very small.

Workaround: The commands in the Printing library (`revPrintField`, `revPrintText`, and `revPrintReport`) have been improved to avoid these problems. If you're using the print command directly to print a card or stack, follow these steps:

1. Choose a printer (or use the answer printer command to have the user choose).
2. If the stack's `formatForPrinting` property is true, set it to false, then back to true.

3. Change the fonts of fields on the card(s) you are printing, if necessary, to fonts that are resident on the printer. (To find out what fonts are available on the current printer, use the `fontNames("printer")` form of the `fontNames` function.)

Issue: Setting the `explicitVars` property to true causes instability in the Revolution development environment.

Workaround: None. This issue is being actively investigated and will be addressed in a future version.

Issue: On Windows systems, the Speech library is not yet compatible with version 5.x of Microsoft's speech software.

Workaround: Use version 4.x instead.

Issue: On OS X systems, the `controlKey` and `optionKey` functions occasionally return false even if the keys are being pressed. This issue is an OS problem.

Workaround: Log out of OS X, then log back in. This clears the key detection.

Issue: On Mac OS and OS X systems, keyboard input in system windows may produce unexpected results, and the keystrokes may go to a different window instead. This issue is an OS problem.

Workaround: Avoid using system windows for operations that require keyboard input, such as typing into fields. If you need to get input from the user from a system window, include a button the user can click to display a dialog box instead.

Issue: Screen-related properties such as the `screenDepth` are set only when the application starts up and are not updated when the system settings are changed.

Issue: Under certain circumstances, the system time fails to respect system settings.

Issue: The `convert` command cannot be used with a negative number of seconds (representing a time before January 1, 1970) on Windows systems.

Issue: When the `lookAndFeel` property is set to `Appearance Manager`, setting a standard button's `hilite` property to false on mouseDown does not change the button's appearance. This problem does not occur on Windows or Unix systems, or on Mac OS or OS X systems when the `lookAndFeel` is not set to `Appearance Manager`, and it occurs only with buttons whose style is set to `standard`.

Issue: File names and folder names longer than 32 characters are not fully supported under OS X. You can use such filenames as part of a file path in commands and functions, but the file path returned from the `ask file`, `answer file`, and `answer folder` commands truncates file names longer than 32 characters.

Issue: The `send to program` command does not yet support the new `eppc://` URL scheme that OS X uses for system addresses. This means that on OS X, you can use the `send to program` command to send Apple Events to programs on the same system, but not to other systems.

Workaround: Use sockets to communicate between systems.

Issue: On OS X systems, there is a problem recording sound using the `mp4` compression format.

Workaround: Use a different compression format.

Issue: On Mac OS and OS X systems, certain controls (scrollbars, standard buttons, radio buttons, checkboxes, option menus) may not print properly if the lookAndFeel property is set to `AppearanceManager`.

Workaround: During printing, replace each affected control with an image of the control. (You can create the image automatically in a handler using the `import snapshot` command.) When printing is complete, delete the images. If the exact appearance of these controls is not critical, you can instead change the lookAndFeel to `Macintosh` during printing, then change it back.

Issue: When using QuickTime, the controller thumb in a player does not always update properly while a movie is playing.

Workaround: Do something that will force the controller to be redrawn: for example, set the player's `showController` property to false in a `preOpenCard` handler and back to true in an `openCard` handler, or the reverse with the `alwaysBuffer` property. Or set the stack's `resizable` property to false.

Issue: If the `alwaysBuffer` of a player that is displaying a QuickTime VR movie is set to true, setting the player's `pan`, `tilt`, or `zoom` properties has no effect.

Workaround: Set the player's `currentTime` property after setting the `pan`, `tilt`, or `zoom`.

Issue: Font, size, and style changes cannot be applied when the text selection is empty (a blinking insertion point) and no objects are selected.

Workaround: To apply a font, size, or style change to text, select the text, then choose the correct command from the Font menu.

Issue: A few of the built-in cursors aren't yet implemented.

Workaround: Create a custom cursor to use instead.

Issue: Scrollbar objects whose style is `scale` or `progress`, when they are oriented vertically, do not have the native appearance when the lookAndFeel property is set to `Macintosh` or `Windows 95`; vertical scale and progress bars always are drawn with the Motif appearance. (This limitation does not apply to scrollbar objects whose style is `scrollbar`, or to horizontal scale bars and progress bars, which are always drawn with the correct native appearance for the current platform. It also does not apply on Mac OS and OS X systems when the lookAndFeel property is set to `AppearanceManager`, which uses the standard system routines for display of user interface objects such as scrollbars.)

Issue: There are cosmetic issues on Mac OS and OS X systems when the `autoArm` property of a standard button is set to true: the button is displayed as a rectangle button instead of standard, and if the button has an icon, it does not disappear when the `armedIcon` appears.

Issue: Rapid changes in stack windows may not be seen on OS X systems because of the way OS X buffers window contents.

Workaround: Use a very short wait command after the change, then continue with the handler.

If you encounter issues or problems not listed above, please report them through the bug reporting system at <http://www.runrev.com/revolution/developers/bugdatabase/>, describing the problem in as much detail as possible. In particular, please include:

- ò the platform and operating system version you were using
- ò the steps that caused the problem to appear
- ò whether the problem is reproducible

Tip: Revolution development environment errors are logged to a file named `sessionlog.txt`, which is erased each time you start up Revolution. The file is located inside the folder `revolution/components/save/`. If you are experiencing problems with the development environment, please include this file along with your bug report.

About what was new in version 2.0-2.0.3

See Also:

About whatÆs new in version 2.1-2.1.1

This topic describes changes to Revolution between version 1.1.1 (April 2002) and 2.0.3 (August 2003).

The information in this topic is the same as the information contained in the ôRead Me Firstö and ôWhatÆs Newö files that shipped with version 2.0.3.

Contents:

Overview of Version 2.0.3

Overview of Version 2.0.2

Overview of Version 2.0.1

Overview of Version 2.0

Where Is It?

New Features

Issues Addressed in This Version

Limitations and Known Issues

Overview of Version 2.0.3

Version 2.0.3 was a bug fix release. There were no new features, changes to Transcript, or user-interface changes.

The engine version for Revolution 2.0.3 was 2.5 build 1.

Overview of Version 2.0.2

Version 2.0.2 was a bug fix release. There were no new features, changes to Transcript, or user-interface changes.

The engine version for Revolution 2.0.2 was 2.5 build 1.

Issues addressed in this version

ò Some problems with the Distribution Builder have been addressed.

ò A problem where preference settings were not recognized has been fixed.

Overview of Version 2.0.1

Version 2.0.1 was a bug fix release. There were no new features, changes to Transcript, or user-interface changes.

The engine version for Revolution 2.0.1 was 2.5 build 1.

Issues addressed in this version

ò The Distribution Builder window now appears with the correct size.

ò Revolution no longer occasionally asks for a disk in Drive D on Windows systems.

ò A problem that caused some stacks saved in beta versions to fail to open in 2.0 (or to cause a CPU spike when opened) has been fixed.

Overview of Version 2.0

Version 2.0 was a feature release. Major new features included:

- ò New Mach-O architecture for OS X standalone applications
- ò Scriptable drag and drop
- ò New text capabilities: Unicode, RTF, & improved regular expressions
- ò XML parser and SOAP support
- ò Table and database field types
- ò Easy-to-use report generator
- ò Video recording using QuickTime or Video for Windows
- ò Text to speech
- ò All-new, all-singing, all-dancing debugger
- ò Transcript Cookbook of sample code recipes, extensively annotated

Other changes included:

- ò New sound recording architecture
- ò Enhanced database access
- ò New FTP functionality
- ò Direct access to the clipboard

In addition, the development environment was extensively improved and updated. OS X users in particular will find that the user interface is more Jaguar-friendly.

Important! Whenever you upgrade to a new version, be sure to make a backup copy of all your stack files before saving them with the new version. This ensures that in case of a problem or incompatibility, you can go back to an older version temporarily until the problem is resolved.

The format of stack files has not changed between version 1.1.1 and 2.0. This means that you can open and use version 2.0 stack files in version 1.1.1 (as long as the stack file does not contain Unicode text), although features that are new for 2.0 will not work in earlier versions.

The engine version for Revolution 2.0 was 2.5 build 1.

New Transcript terms

/**/, acceptDrop, allowInlineInput, answer record, backslash, boundingRect, clipboardData, constantNames, crop, defaultCursor, deleteRegistry, DNSServers, dragData, dragDestination, dragDrop, dragEnd, dragEnter, dragLeave, dragMove, dragSource, dragStart, driverNames, dropChunk, gRevAppIcon, gRevProfileReadOnly, gRevSmallAppIcon, left, libURLDownloadToFile, libURLftpCommand, libURLftpUploadFile, libURLSetFTPListCommand, libURLSetStatusCallback, lineDelimiter, LPT1:, null, recordChannels, recordCompression, recordCompressionTypes, recordFormat, recordInput, recordRate, recordSampleSize, recursionLimit, resume, revAddXMLNode, revAppendXML, revCloseCursor, revCloseDatabase, revCloseVideoGrabber, revCommitDatabase, revCreateXMLTree, revCreateXMLTreeFromFile, revDeleteAllXMLTrees, revDeleteXMLNode, revDeleteXMLTree, revEndXMLNode, revEndXMLTree, revExecuteSQL, revInitializeVideoGrabber, revIsSpeaking, revLicenseType, revMail, revMoveToFirstRecord, revMoveToLastRecord,

revMoveToNextRecord, revMoveToPreviousRecord, revPreviewVideo, revPutIntoXMLNode, revRecordVideo, revResizeStack, revRollBackDatabase, revSetCardProfile, revSetDatabaseDriverPath, revSetSpeechPitch, revSetSpeechSpeed, revSetSpeechVoice, revSetStackFileProfile, revSetStackProfile, revSetVideoGrabberRect, revSetVideoGrabSettings, revSetXMLAttribute, revSpeak, revSpeechVoices, revStartXMLData, revStartXMLNode, revStartXMLTree, revStopPreviewingVideo, revStopRecordingVideo, revStopSpeech, revUnloadSpeech, revVideoFrameImage, revVideoGrabDialog, revVideoGrabIdle, revVideoGrabSettings, revXMLAddDTD, revXMLAttribute, revXMLAttributes, revXMLAttributeValues, revXMLChildContents, revXMLChildNames, revXMLFirstChild, revXMLMatchingNode, revXMLNextSibling, revXMLNodeContents, revXMLNumberOfChildren, revXMLParent, revXMLPreviousSibling, revXMLRootNode, revXMLText, revXMLTree, revXMLTrees, revXMLValidatedDTD, right, RTFText, selectedImage, sheet, slash, suspend, unicodeText, useUnicode, windowShape

Changed and extended Transcript terms

abbreviated, angle, answer, answer file, answer folder, ask, ask file, ask password, backgroundColor, borderColor, charToNum, clipboard, close driver, close process, copy, currentTimeChanged, externalCommands, externalFunctions, externals, ftp, go, hGrid, hiliteColor, htmlText, long, numToChar, open driver, open printing, open process, owner, popup, queryRegistry, read from driver, read from process, record sound, repeat, return, revCurrentRecord, revCurrentRecordIsFirst, revCurrentRecordIsLast, revDatabaseColumnCount, revDatabaseColumnIsNull, revDatabaseColumnLengths, revDatabaseColumnNamed, revDatabaseColumnNames, revDatabaseColumnNumbered, revDatabaseColumnTypes, revDatabaseConnectResult, revDatabaseCursors, revDatabaseID, revDatabaseType, revDataFromQuery, revGoURL, revNumberOfRecords, revOpenDatabase, revOpenDatabases, revPrintText, revQueryDatabase, revQueryDatabaseBLOB, revQueryResult, screenMouseLoc, serialControlString, setProp, setRegistry, shell, short, signal, specialFolderPath, stderr, stdin, stdout, textFont, uniDecode, uniEncode, vGrid, write to driver, write to process

Removed Transcript terms

recordFormats (supplanted by recordCompressionTypes)

Where Is It?

Several features have been moved or renamed in this version. HereÆs a guide to whatÆs where for users of Revolution 1.1.1 moving to 2.0:

Properties palette:

Now called the property inspector. Instead of tabs, the property inspector has several panes that are accessed with a menu at the top of the inspector.

Script editor:

Now a separate window, not part of the Properties palette. Because the script editor is now a window instead of a palette, you can switch between script windows and stack windows easily. You open the script editor by choosing Object menu Object Script, Object menu Card Script, or Object menu Stack Script, or using the Script button in the Toolbar, or the contextual menu shortcut.

Profile Manager:

Moved to the Property Profiles pane of the property inspector.

Geometry Manager:

Moved to the Geometry pane of the property inspector.

Animation Manager:

Now called the Animation Builder.

Menu Manager:

Now called the Menu Builder.

Database Manager:

Replaced by the Database Query Builder, which creates automatic queries that can be linked to fields to display the data.

Text palette:

Moved from the Object menu to the Text Formatting pane of the property inspector.

Colors and Patterns palette:

Moved from the Object menu to the Colors & Patterns pane of the property inspector.

Inks palette:

Moved from the Object menu to the Inks pane of the property inspector.

Alignment palette:

Moved from the Object menu to the property inspector. Select the objects you want to align, then display the property inspector and choose "Align Objects" from the menu at the top of the inspector.

Script Debug window:

The debugger is no longer a separate mode with its own window. Instead of entering Script Debug mode, you set a breakpoint in a handler you want to debug by clicking next to the line where you want the breakpoint. When the line with the breakpoint is executed, or when you click "Debug" in the error window, the script editor opens, displaying debugging options that let you execute the handler line by line.

Application Overview:

Moved to Tools menu Application Browser. The user interface has been significantly streamlined.

Control-click (Mac OS or OS X) or right-click (Unix or Windows) the items in the Application Browser to get access to additional features.

New Features

This section describes the new features introduced in version 2.0 in detail.

New OS X architecture

The Carbon (OS X) engine is now built as a Mach-O executable. For the most part, this change is internal only and does not affect developers. There are a few areas where you may notice effects of this change:

• OS X Revolution applications (including Revolution itself) can no longer optionally run under Mac OS using CarbonLib. (In previous versions, applications compiled for OS X could run under Mac OS, although this was not recommended.)

ò OS X externals must be recompiled, since the new engine does not support calling CODE resources.

ò Because OS X applications are now delivered as application bundles, the filename of stack property reports the path to the executable inside the bundle, rather than the bundle itself.

ò The format of absolute pathnames on OS X systems has changed. Instead of starting at the desktop, the folder structure now starts at the startup disk. The path of a file on the startup disk starts with a folder name, not with the startup disk's name. Pathnames of files on other disks start with `/Volume/` followed by the disk name. (For more information, see the documentation topic `About file specifications and file paths`.)

ò Several issues affecting OS X compatibility have been addressed. In particular, the shell function and the open process, read from process, write to process, and close process commands now work properly on OS X. The screenMouseLoc property and the signal message, and the stdin, stdout, and stderr keywords are now supported on OS X.

Scriptable drag and drop

Earlier versions of Revolution supported dragging and dropping text selections within a field. Version 2.0 adds support for dragging text selections between fields without any scripting needed.

There is also an extensive new script architecture for specifying the format and content of dragged data, whether an object allows dragging, and whether an object can accept a drop. You can drag any type of data from any control to any other control, drag between a Revolution application and another application, and override the automatic field dragging behavior if desired.

New text capabilities

Text in fields can now be stored and displayed as Unicode, allowing entry and viewing of text in languages (such as Japanese) that do not use the Roman alphabet and require double-byte characters.

ò Users can enter Unicode text in fields using their operating system's standard tools for double-byte text entry.

ò The new unicodeText property lets you set, get, and store Unicode field contents and import or export Unicode text files.

ò The htmlText property has been updated to support Unicode.

ò You can specify a language with the uniDecode and uniEncode functions to convert between Unicode text and language-specific encodings such as Shift-JIS. These functions no longer assume a big-endian architecture, which means they now work properly on all platforms.

ò The numToChar and charToNum functions support Unicode if the useUnicode property is true.

Note: The character keyword, used in chunk expressions, still refers to a single-byte character, even if the text being manipulated is Unicode.

ò The new RTFText property lets you set, get, and store styled text in RTF format, and import or export RTF files. (Only styles supported by Revolution fields are recognized in imported RTF.)

ò Regular expressions (used in the `matchText`, `matchChunk`, and `replaceText` functions) have been extended and are now fully Perl-compatible.

XML library

A suite of commands, messages, and functions for parsing XML files and creating and manipulating XML content is now available.

ò The `revCreateXMLTree` function creates a data structure from XML data.

ò The `revCreateXMLTreeFromFile` function parses an XML file, optionally creating a tree structure and optionally sending messages when it encounters a new XML element during parsing. Since this function does not need to read the entire contents of the file into memory at once, you can use it to work with XML documents larger than available memory.

ò Other commands and functions in the XML library allow traversing the nodes of an XML tree, validating an XML document against a DTD, adding and deleting nodes and content, converting an XML tree to XML text, and more.

ò For an extensive example of SOAP, see the file `SOAP Toolbox.rev` in the Revolution folder.

New field capabilities

Fields can now display tab-delimited data in spreadsheet form.

ò If a field's `vGrid` property is set to true and its `tabStops` is not empty, any tab-delimited chunks in the field are truncated at the next tab stop. This lets fields support a spreadsheet display of data.

ò The lines drawn by the `vGrid` and `hGrid` properties are now displayed in the field's `borderColor` instead of the `hiliteColor`.

ò You can use the Table pane in a field's Property Inspector to easily set up the field to display data in table form.

Report Builder

Revolution now includes a built-in report generator.

ò Lay out a custom report on a card, using any objects, and use report viewer objects to display a field's content for each page of the report.

ò The Report Builder specifies which cards to print (all, a range, marked or unmarked cards, or all cards selected by an expression), the printing format, and a header and footer.

ò The new `revPrintReport` command uses the settings from the Report Builder.

Video recording library

The new Video library offers a complete set of commands and functions for capturing video.

ò You can use Video for Windows or QuickTime video capture.

ò You can display the system settings dialog boxes, store and restore video settings, display a preview window on the screen showing the input from the video source, and capture video input to a file.

Text to speech library

The commands and functions in the new Speech library let you have the computer speak text from any source, of any length, out loud. You can control the voice used, speed, pitch, and volume.

Debugger

The debugger has been greatly enhanced, with a new interface, new features, and script control.

ò Message Watcher lets you see which messages are being sent as they are dispatched.

ò Variable Watcher lets you see and change variable values during execution.

To enter the debugger, click ôDebugö in the Error window.

To force a handler to enter the debugger, click the line of the script where you want the debugger to break in, then choose Debug menu Set Breakpoint.

New sound recording architecture

The record sound command has been revamped and simplified.

ò Functionality formerly accessed with various parameters of the record sound command has been moved to the new answer record command and recordCompression, recordRate, and recordSampleSize properties. Use the recordCompressionTypes property, which replaces the recordFormats property, to get a list of available codecs for the recordCompression property.

ò You can now specify the file format of recorded sounds using the recordFormat property. In previous versions, sounds were always recorded in QuickTime format.

ò You can now specify the sound input source with the recordInput property. You can also use the recordChannels property to record in stereo or mono.

Database enhancements

ò Compatibility with PostgreSQL has been added to the Database library.

ò MySQL database connections now support SSL.

ò Several synonyms have been introduced for functions in the Database library, and some functions have been changed to commands. The old function names are still available, so your existing scripts will continue to work.

ò The Database library can now be used from Revolution CGIs and shell scripts.

ò For Valentina databases on OS X systems, the revOpenDatabase command now uses a standard Revolution file path instead of a Mac OS file path.

Internet library and other Internet additions

ò The new `libURLftpDownloadToFile` and `libURLftpUploadFile` commands allow transferring files (even files too large to fit in available memory).

ò The new `libURLftpCommand` command sends any FTP command to an FTP server.

ò The new `libURLSetStatusCallback` command allows status messages to be sent during any upload or download. Use this command to display a status message or progress bar during file transfers.

ò You can now specify a port number in an ftp URL.

ò ftp URLs now work with directory names that include spaces.

ò The new `DNSServers` function returns a list of all configured DNS servers.

ò The new `revMail` command opens a new email message in the user's default email program.

ò The `revGoURL` command now allows you to specify a default browser for Unix systems.

Clipboard improvements

ò Copying and pasting styled text and images between applications is now supported.

ò The new `clipboardData` property lets you see and change the data on the clipboard: text, styled text, bitmapped images, or files.

Image manipulation changes

ò You can now non-destructively rotate an image using the `angle` property.

ò The new `crop` command changes the size of an image by cropping it or adding blank space around it, instead of by scaling it.

ò You can now copy a portion of an image that's selected with the Select tool. The `selectedImage` function reports which image has the current selection.

ò You can now copy an image or portion of an image and paste it into another application.

ò The new `windowShape` property lets you specify a clipping shape for a stack window, based on the mask of any image.

Documentation changes and additions

ò The documentation now includes the Transcript Cookbook of code examples. To see the Cookbook, click "Transcript Cookbook" in the main Documentation window. Or choose any topic starting with "Recipe for..." from the Development Guide section of the documentation, or from the See Also menu in any documentation topic.

Click the "Demo" button at the top of the Cookbook window to see a demonstration stack illustrating the handler (or handlers) described in the current topic.

ò By popular demand, you can now search the text of the Transcript Dictionary from its page in the main documentation window. Follow these steps:

1. Go to the Transcript Dictionary page of the main documentation window.
2. Choose ⌘Search for⌘ from the menu at the left of the text box.
3. Enter the text you want to search for, then press Return or Enter. The list shows the terms where the search text appears.

You can also search in the Transcript Cookbook, Troubleshooting, and Glossary pages of the main documentation window.

ò You can now click a menu item that's referred to in the documentation to open its topic in the Menu Reference.

ò You can now select text in the documentation and copy it without holding down the Option key or Alt key.

Miscellaneous Transcript changes and additions

ò New suspend and resume messages are now sent when the application goes to the background or comes to the foreground.

ò The new defaultCursor property lets you replace the browse (hand) cursor with any built-in or custom cursor.

ò The popup command now lets you specify the popup menu's location.

ò You can now set profiles for all the objects in a card, stack, or stack file at once using the revSetCardProfile, revSetStackProfile, and revSetStackFileProfile commands.

ò You can use the new deleteRegistry function to delete a registry entry. The setRegistry and queryRegistry functions now let you specify and get the data type of an entry. The setRegistry function can now be used to delete a subkey.

ò The specialFolderPath function now supports system constants (on Mac OS and OS X) and CSIDL numbers (on Windows), giving access to a wider range of special folders.

ò The new revLicenseType function specifies which license type the development environment is running under.

ò The currentTimeChanged message now includes a parameter giving the new time.

ò The revPrintText command now supports specifying default font attributes.

ò The new constantNames function lists all built-in constants. The slash, backslash, left, right, LF, and null constants have been added.

ò The new lineDelimiter property (similar to the itemDelimiter) lets you change the line delimiter to any character.

- ò You can now get an object's long, short, or abbreviated owner.
- ò The repeat control structure now allows incrementing the count variable by a number other than 1: repeat with x = 1 to 10 step 2
- ò It is no longer possible to explicitly change the variable inside a loop of the form repeat for each chunkType chunkVariable in container.
- ò The new recursionLimit property is useful for debugging and for increasing Transcript's ability to handle deeply nested recursive functions.
- ò Setting a custom property inside a setProp handler now sends a setProp trigger, unless you're setting the handler's own custom property.
- ò Block comments (multiple-line comments) are now supported within `/**/`.

Miscellaneous other changes and additions

- ò Double-clicking a stack file now loads the development environment in addition to opening the stack. To launch Revolution without loading the development environment (for testing purposes), double-click the stack file while holding down the Command key (Mac OS and OS X) or Control key (Unix and Windows).
- ò You can now build OS X standalones on any platform.
- ò The script editor now opens in a modeless window instead of a palette. This means that you can now click a stack window to move it in front of a script window, and go back and forth between them easily.
- ò The Properties palette has been extensively reworked (and is now called the property inspector).
- ò When saving a stack for the first time without having set its name property, you will now be prompted to name the stack (as well as the stack file).
- ò Revolution now supports mouse wheels on OS X. The mouse wheel can be used to scroll a field, and it sends a rawKeyDown message.
- ò On Mac OS and OS X systems, placing a backslash (before menu metacharacters such as `ô` now escapes those characters, so that they are shown as-is in the menu. (This does not apply to pulldown menus that are displayed in a window.)
- ò Revolution is no longer compatible with OpenLook Window Manager.

Note: The Starter Kit Edition is now called the Free Edition. You can configure the Free Edition as an evaluation copy, with no script-length limits, for 30 days. (After the evaluation period, unlicensed copies will continue to function with Free Edition script-length limits. Purchasing a license removes these limits.)

Issues Addressed in This Version

Compatibility issues

ò OS X applications now automatically delete the last two lines in the first menu of a stack's menu bar. This is to prevent duplication of the "Quit application" menu item in the OS X Application menu. (Normally, the first menu is the File menu, and the last two menu items are a separator line and "Quit".) This lets you use the same menu bar for all platforms without breaking user-interface guidelines for OS X, which call for the "Quit" menu item to be in the Applications menu and not the File menu.

ò Default buttons now throb.

ò Sheets are now supported using the ask, answer, and related commands. Custom sheets are supported using the go and sheet commands.

Development environment issues

ò Choosing File menu Import As Control Snapshot now stacks the windows correctly when they reappear.

ò The text highlight color is now correctly displayed in the script editor and the ask dialog box.

ò The property inspector no longer insists on setting the tabGroupBehavior to true, and properly updates the acceleratorModifiers property.

ò A problem where uncommenting script lines sometimes deleted characters that were part of the line has been fixed.

ò The Image Chooser now displays images from groups that aren't placed on any card, and displays grouped images only once, instead of one time for each card the group is placed on.

Other issues

ò File paths longer than 255 characters are now fully supported on Mac OS and OS X systems.

ò Undo behavior has been improved: text is no longer mangled when undoing after selecting text and replacing it, and styles are no longer lost when Undo restores cut or deleted text.

ò Parameters in an on, setProp, getProp, or function control structure can no longer have the same name as a built-in property or function.

ò The mouse function now reports the state of the mouse button at the moment the function is called. Formerly, this function reported whether the mouse button had been pressed since the current handler started. (In general, it is recommended to handle the mouseDown, mouseRelease, and mouseUp messages rather than check the state of the mouse in a handler.)

ò The print command no longer prints the menubar area of stacks whose editMenus property is set to false.

ò Selections in an image object can now be copied.

ò On OS X systems, Revolution now uses Quartz routines to draw antialiased text.

ò A problem that caused the imageData property to report values in the wrong order on Windows systems has been fixed. The imageData is now returned in the same order on all platforms.

ò Revolution now sends mouseEnter and mouseLeave messages to objects that are shown or hidden while the mouse pointer is over them, and to objects when a handler moves them underneath the current mouse position.

ò Attempting to connect to a database when no corresponding database driver is found no longer causes a crash.

ò Various issues that were causing problems with socket timing have been addressed.

ò setProp triggers and getProp calls now go through frontScripts, as well as the rest of the message path.

Limitations and Known Issues

Please report other bugs or issues to support@runrev.com.

Issue: Use of the shell function on Windows systems with Norton AutoProtect installed may cause a crash.

Workaround: If you need to use the shell function on Windows systems, disable Norton AutoProtect.

Issue: On Mac OS systems, the InputSprocket extensions have been reported to cause instability with some applications, including Revolution.

Workaround: If you experience instability and you do not require the sprocket extensions, try turning these extensions off in the Extensions Manager control panel.

Issue: Some Windows graphics drivers have compatibility problems with Revolution, in particular with images. The primary symptom is that images are drawn in the wrong colors.

Workaround: If you see this symptom, make sure you have the latest drivers from the vendor of your graphics card, and try turning off graphics acceleration (in the Performance tab in the System Control Panel). If this does not fix the problem, please report the bug to your graphics card vendor.

Issue: Some Windows versions show font problems when printing text. In particular, Windows XP printouts may have blank lines between lines of text or half-length printed lines. Printed text on Windows ME or Windows 98 systems may be very small.

Workaround: The commands in the Printing library (revPrintField, revPrintText, and revPrintReport) have been improved to avoid these problems. If youÆre using the print command directly to print a card or stack, follow these steps:

1. Choose a printer (or use the answer printer command to have the user choose).
2. If the stackÆs formatForPrinting property is true, set it to false, then back to true.
3. Change the fonts of fields on the card(s) you are printing, if necessary, to fonts that are resident on the printer. (To find out what fonts are available on the current printer, use the fontNames("printer") form of the fontNames function.)

Issue: Setting the explicitVars property to true causes instability in the Revolution development environment.

Workaround: None. This issue is being actively investigated and will be addressed in a future version.

Issue: The diskSpace function may return an incorrect value if the disk is larger than 1G.

Workaround: None.

Issue: On Windows systems, the Speech library is not yet compatible with version 5.x of Microsoft's speech software.

Workaround: Use version 4.x instead.

Issue: The controlKey and optionKey functions occasionally return false on OS X even if the keys are being pressed. This issue is an OS problem.

Workaround: Log out of OS X, then log back in. This clears the key detection.

Issue: Screen-related properties such as the screenDepth are set only when the application starts up and are not updated when the system settings are changed.

Workaround: None.

Issue: Under certain circumstances, the system time fails to respect system settings.

Workaround: None.

Issue: The convert command cannot be used with a negative number of seconds (representing a time before January 1, 1970) on Windows systems.

Workaround: None.

Issue: When the lookAndFeel property is set to 'Appearance Manager', setting a standard button's hilite property to false on mouseDown does not change the button's appearance. This problem does not occur on Windows or Unix systems, or on Mac OS or OS X systems when the lookAndFeel is not set to 'Appearance Manager', and it occurs only with buttons whose style is set to 'standard'.

Workaround: None.

Issue: File names and folder names longer than 32 characters are not fully supported under OS X. You can use such filenames as part of a file path in commands and functions, but the file path returned from the ask file, answer file, and answer folder commands truncates file names longer than 32 characters.

Issue: The send to program command does not yet support the new 'eppc:' URL scheme that OS X uses for system addresses. This means that on OS X, you can use the send to program command to send Apple Events to programs on the same system, but not to other systems.

Workaround: Use sockets to communicate between systems.

Issue: On OS X systems, there is a problem recording sound using the 'mp4a' compression format.

Workaround: Use a different compression format.

Issue: The titleWidth property does not function correctly if the lookAndFeel is set to 'Appearance Manager'. In this case, the title of a menu button with a nonzero titleWidth appears inside the button instead of beside it.

Workaround: Instead of using the titleWidth, create a locked field to hold the title and place it beside the menu button.

Issue: On Mac OS and OS X systems, certain controls (scrollbars, standard buttons, radio buttons, checkboxes, option menus) may not print properly if the lookAndFeel property is set to 'Appearance Manager'.

Workaround: During printing, replace each affected control with an image of the control. (You can create the image automatically in a handler using the import snapshot command.) When printing is complete, delete the images. If the exact appearance of these controls is not critical, you can instead change the lookAndFeel to `Macintosh` during printing, then change it back.

Issue: When using QuickTime 5, the controller thumb in a player does not always update properly while a movie is playing. This issue is being actively investigated with Apple.

Workaround: Do something that will force the controller to be redrawn: for example, set the player's `showController` property to false in a `preOpenCard` handler and back to true in an `openCard` handler, or the reverse with the `alwaysBuffer` property. Or set the stack's `resizable` property to false.

Issue: If the `alwaysBuffer` of a player that is displaying a QuickTime VR movie is set to true, setting the player's `pan`, `tilt`, or `zoom` properties has no effect.

Workaround: Set the player's `currentTime` property after setting the `pan`, `tilt`, or `zoom`.

Issue: Omitting a parameter in the `revMail` command causes the remaining parameters to be ignored by the mail program.

Workaround: Use a space character as a parameter instead of leaving the parameter empty.

Issue: Font, size, and style changes cannot be applied when the text selection is empty (a blinking insertion point) and no objects are selected.

Workaround: To apply a font, size, or style change to text, select the text, then choose the correct command from the Font menu.

Issue: A few of the built-in cursors aren't yet implemented.

Workaround: Create a custom cursor to use instead.

Issue: Scrollbar objects whose style is `scale` or `progress`, when they are oriented vertically, do not have the native appearance on Mac OS and Windows systems; vertical scale and progress bars always are drawn with the Motif appearance. (This limitation does not apply to scrollbar objects whose style is `scrollbar`, or to horizontal scale bars and progress bars, which are always drawn with the correct native appearance for the current platform. It also does not apply on Mac OS and OS X systems when the `lookAndFeel` property is set to `Appearance Manager`, which uses the standard system routines for display of user interface objects such as scrollbars.)

Issue: There are cosmetic issues on Mac OS and OS X systems when the `autoArm` property of a standard button is set to true: the button is displayed as a rectangle button instead of standard, and if the button has an icon, it does not disappear when the `armedIcon` appears.

A few cosmetic issues remain on OS X systems. These include the following:

Issue: In modal dialog boxes, mouse clicks are blocked if a tool tip is visible.

Issue: The interactive form of the import snapshot command does not display a `marching ants` rectangle.

Issue: When opening a stack whose `windowShape` property is not zero, there is a brief white flash.

Issue: Setting the `hidePalettes` property to false does not work: palettes are always hidden when the application is in the background.

Issue: Rapid changes in stack windows may not be seen on OS X systems because of the way OS X buffers window contents. As a workaround to ensure that a brief change can be seen, use a very short wait command after the change, then continue with the handler.

If you encounter issues or problems not listed above, please report them through the bug reporting system at <<http://www.runrev.com/Revolution1/bugzilla/>>, describing the problem in as much detail as possible. In particular, please include:

- ò the platform and operating system version you were using
- ò the steps that caused the problem to appear
- ò whether the problem is reproducible

Revolution development environment errors are logged to a file named `sessionlog.txt`, which is erased each time you start up Revolution. The file is located inside the folder `revolution/components/save/`. If you are experiencing problems, please send this file to support@runrev.com along with your bug report.

About what was new in version 1.1.1

See Also:

About what's new in version 2.1-2.1.1

This topic describes changes to Revolution between version 1.1 (November 2001) and 1.1.1 (April 2002).

The information in this topic is the same as the information contained in the "Read Me First" file that shipped with version 1.1.1.

Contents:

Overview of Version 1.1.1

New Features

Issues Addressed in This Version

Limitations and Known Issues

Overview of Version 1.1.1

Version 1.1.1 was primarily a maintenance release, addressing issues that affected stability for some developers. In addition, the following features were added:

- ò Extensions and additions to the Internet library
- ò Ability to communicate with arbitrary devices
- ò New file manipulation commands
- ò New database support including Valentina database support on Mac OS , OS X, and Windows systems and MySQL support for Mac OS X systems.

Important! Whenever you upgrade to a new version, be sure to make a backup copy of all your stack files before saving them with the new version. This ensures that in case of a problem or incompatibility, you can go back to an older version temporarily until the problem is resolved.

The engine version for Revolution 1.1.1 was 2.4.1 build 7.

New Transcript terms

close driver, dontUseQTEffects, libURLErrorData, libURLftpUpload, libURLSetFTPMode, libURLSetFTPStopTime, libURLSetLogField, libURLVersion, open driver, read from driver, revCopyFile, revCopyFolder, revdb_columnisnull, revdb_columntypes, revdb_dbtype, revdb_querylist, revdb_queryblob, revDeleteFolder, revMoveFolder, revRotatePoly, write to driver

Changed and extended Transcript terms

answer color, cachedURLs, htmlText, http, launch, open process, rename, revdb_connect, URLStatus

New Features

This section describes in detail all new features introduced in version 1.1.1.

Expanded Internet support

You can now use the `URLStatus` function to monitor both uploads and downloads via `http` or `ftp` protocols. If an error occurs while uploading or downloading a file with `get` or `put`, the error message now appears in the result function. The `libURLErrorData` function returns detailed information on any errors encountered during the most recent transfer.

The `cachedURLs` function now returns only URLs that have been successfully downloaded.

The mode (active or passive) of FTP transfers can be set with the new `libURLSetFTPMode` command. The `libURLftpUpload` command allows non-blocking uploads to an FTP server. Use of the `load` command for both `http` and `ftp` downloads is now fully supported.

Device driver support

The new `open driver`, `read from driver`, `write to driver`, and `close driver` commands can be used to control peripheral devices, get data, and send data through communications peripherals that are not connected to the standard serial ports.

New file system features

Deleting folders has been made easier with the new `revDeleteFolder` command, which can remove folders that are not empty. The new `revCopyFolder` and `revCopyFile` commands copy all attributes of files and folders, including Mac OS resource forks. The `revMoveFolder` command moves a folder and its contents to a new location.

The `rename` command now can be used to move a file or folder on Mac OS and Windows systems in addition to changing its name.

Additional database support

You can now use the `revdb_connect` function to connect to Valentina databases, in addition to MySQL, Oracle, and ODBC databases. Valentina is a fast, efficient single-user database engine for Mac OS, OS X, and Windows systems. A demonstration version of Valentina (with a ten-minute connection limit) is included in Revolution distributions for these platforms and is located inside the Revolution folder.

The Valentina database engine can be licensed from Paradigma Software. For more information about Valentina, visit the Paradigma web site at <http://www.paradigmasoft.com/product/vxcmd.html>.

The new `revdb_columntypes` and `revdb_columnisnull` functions can be used to test the data in a specified database field. The new `revdb_queryblob` function can handle BLOB data objects containing large amounts of binary data. The new `revdb_querylist` function can be used to fetch data from a database into a variable. The new `revdb_dbtype` function can be used to find out what kind of database a connection ID is associated with.

The contents of the Database Manager's `Help` tab has been moved to the Revolution Encyclopedia for this release. Look in the documentation for the article titled `About connecting to and using SQL databases`.

Miscellaneous Transcript changes and additions

• The new `dontUseQTEffects` property can be used to constrain visual effects to use the built-in routines instead of QuickTime. This addresses an issue where QuickTime-based visual effects were cosmetically unsatisfactory on slower systems.

- ò The new revRotatePoly command rotates a line, curve, or polygon graphic through a specified angle.
- ò The answer color command now takes an optional default color.
- ò The htmlText property now uses `<a> ` to enclose text whose linkText property is not empty.

Miscellaneous other changes and additions

Important! The script editor's behavior has changed. It no longer asks whether to save changes when you close the script editor. If you close without applying changes, any changes will be lost.

- ò Progress is now shown in the Distribution Builder when downloading a needed engine.
- ò On Windows systems, the `.rev` extension is now automatically registered to Revolution.
- ò The limit on the number of stacks shown in the `Recent Stacks` submenu in the File menu has been increased.
- ò The `Development Guide` page of the main Documentation window now shows a category-based listing of all topics in the documentation. The `Encyclopedia` page has been removed. (All Encyclopedia topics can now be found on the `Development Guide` page.)
- ò You can now search the full text of a documentation section, instead of only the topic titles, by holding down the Option or Alt key while pressing Return in the Find box at the top of the window.

Issues Addressed in This Version

Compatibility issues

- ò Compatibility of the Carbon (OS X) engine with Mac OS is greatly improved. It is still recommended to use the Mac OS engine on Mac OS systems for best results. If you want to try using the Carbon distribution of Revolution on Mac OS 8.0 or later, make sure you have the latest version of CarbonLib installed in the Extensions folder.
- ò Problems that caused the PPC (Mac OS) engine to crash on Mac OS 7.6 and earlier have been resolved.
- ò Problem causing menu items in Image Library to be inappropriately disabled on Mac OS systems has been fixed.
- ò A crash bug that showed up when using tool tips on OS X systems has been terminated with extreme prejudice.
- ò Development environment cosmetic glitches on OS X (mostly involving option menus) have been fixed.
- ò Problem where short menus sometimes displayed incorrectly on OS X systems has been fixed.
- ò On Unix systems, the Colors palette has improved stability.
- ò Exceeding the Starter Kit limits under Windows NT no longer causes a crash.

ò The Database library now works under OS X.

ò Problem that could cause crashes on Windows systems if you quit with open database connections has been fixed.

Development environment issues

ò Issue that caused the Application Overview and message box to open offscreen after changing monitor resolution has been fixed.

ò Closing a stack after saving no longer asks to save changes again.

ò Using the contextual menu for a control no longer sends a mouseUp message to the control.

ò Cutting text in the script editor is now considered a change and enables the Apply button.

ò Cosmetic problems when scrolling in Icon Chooser have been fixed.

ò Problem where nudge arrows in Alignment and Basic Properties occasionally stayed pressed after the mouse button was released, causing the selected control to fly uncontrollably across the stack window and out of sight, has been addressed.

ò Application Overview now displays and updates objects more reliably, in particular nested groups and groups being edited.

ò Properties palettes have several fixes, update more reliably, and have improved stability.

ò ⌘New button on the Toolbar now works more reliably.

ò Message box now resizes properly under more circumstances.

Distribution Builder issues

ò Distribution Builder no longer unexpectedly deletes files.

ò Distribution Builder now respects the name of the data folder you set.

ò Building a standalone application no longer leaves an extraneous file in the folder.

ò Distribution Builder now builds multiple distributions more reliably on Unix systems.

Printing issues

ò The revPrintField command now handles tab-delimited text in fields correctly.

ò Issues when printing multiple pages from the documentation have been fixed.

Other issues

ò The Support email form has been updated.

ò EPS objects created with the create command are no longer enormous by default.

- ò A group placed with the place command when the Browse tool is being used is no longer automatically selected.
- ò Import of HyperCard stacks has been further improved for greater stability.
- ò On Windows systems, the menu bar can no longer be maximized.
- ò Script error reporting, particularly when encountering compile errors, is improved.
- ò The dialog boxes used for the answer and ask commands now lock messages. This addresses a problem where using these commands in handlers that respond to a focus change (such as openField) sometimes produced unexpected results.

Limitations and Known Issues

Please report other bugs or issues to support@runrev.com.

- ò Font, size, and style changes cannot be applied when the text selection is empty (a blinking insertion point) and no objects are selected. To apply a font, size, or style change to text, select the text, then choose the correct command from the Font menu.
- ò Copying and pasting between programs supports only plain text, and styles are lost. (Copying and pasting text within Revolution preserves styles.) To work around this limitation, see the `htmlText` property in the Transcript Dictionary section of the documentation.
- ò Selections in an image object cannot be copied (although you can copy and paste an image object as a whole within Revolution). This issue will be resolved in the near future.
- ò Setting the `explicitVars` property to true will cause instability in the Revolution development environment. This issue is being actively investigated and will be addressed in a future version.
- ò Scrollbar objects whose style is `ôscaleö` or `ôprogressö`, when they are oriented vertically, do not have the native appearance on Mac OS and Windows systems; vertical scale and progress bars always are drawn with the Motif appearance. (This limitation does not apply to scrollbar objects whose style is `ôscrollbarö`, or to horizontal scale bars and progress bars, which are always drawn with the correct native appearance for the current platform. It also does not apply on Mac OS and OS X systems when the `lookAndFeel` property is set to `ôAppearance Managerö`, which uses the standard system routines for display of user interface objects such as scrollbars.)
- ò Recording a sound using the record sound command may produce unexpected results (such as a garbled sound file) if using the 22KHz (`ôbetterö`) sound quality with some versions of QuickTime. This issue is being actively investigated.
- ò Double-clicking a Revolution stack file to launch the Revolution application will open the stack without loading the development environment (which includes Revolution's menu bar). This behavior allows you to test your stack outside the development environment before you build a standalone application from it, but means you cannot edit stacks opened in this way because the development environment is not present. This issue is being actively investigated and will be addressed in a future version.

- ò Revolution does not fully support double-byte character sets (such as the ones used for Asian languages). This issue is being actively investigated and double-byte capability will be fully functional in a future version.
 - ò Choosing File menu Import As Control Snapshot causes the windows to be incorrectly displayed when they reappear.
 - ò The mouse function does not always correctly report the state of the mouse button at the time itÆs called. This issue will be addressed in a future version. In general, it is recommended to handle the mouseDown and mouseUp messages rather than check the state of the mouse in a handler.
 - ò On Mac OS and OS X systems, file paths are limited to 255 characters. This issue will be addressed in a future version.
 - ò On Mac OS systems, the InputSprocket extensions have been reported to cause instability of some applications, including Revolution. If you experience instability and you do not require the sprocket extensions, try turning these extensions off in the Extensions Manager control panel.
 - ò Some Windows graphics drivers have compatibility problems with Revolution, in particular with images. The primary symptom is that images are drawn in the wrong colors. If you see this symptom, make sure you have the latest drivers from the vendor of your graphics card, and try turning off graphics acceleration (in the Performance tab in the System Control Panel). If this does not fix the problem, please report the bug to your graphics card vendor.
 - ò When using QuickTime 5, the controller thumb in a player does not always update properly while a movie is playing. The workaround is to do something that will force the controller to be redrawn: for example, setting the playerÆs showController property to false in a preOpenCard handler and back to true in an openCard handler, or the reverse with the alwaysBuffer property. This problem also does not appear if the stackÆs resizable property is set to false. This issue is being actively investigated with Apple.
 - ò The shell function and open process command are not yet supported when using the OS X engine. (They are supported using the Darwin engine.) The send to program command is also not yet supported, but you can use the do as appleScript form of the do command as a workaround.
 - ò In OS X, the accept datagram form of the accept command ignores datagrams received after the first one. This issue will be fixed in the next version.
 - ò The new open driver syntax is not yet supported on OS X.
- A few cosmetic issues remain on OS X systems. These include the following:
- ò Default buttons do not ôthrobö.
 - ò Sheet dialogs are not yet supported.
 - ò Text is drawn with QuickDraw, so text may look slightly different in Revolution and Revolution applications.
 - ò In modal dialog boxes, mouse clicks are blocked if a tool tip is visible.
 - ò The interactive form of the import snapshot command does not display a ômarching antsö rectangle.

ò Menu buttons whose menuMode is ôpulldownö do not have the Aqua appearance. (This does not apply to menus displayed in the menu bar.)

If you encounter issues or problems not listed above, please report them by email to support@runrev.com, describing the problem in as much detail as possible. In particular, please include:

- ò the platform and operating system version you were using
- ò the steps that caused the problem to appear
- ò whether the problem is reproducible

Revolution development environment errors are logged to a file named ôsessionlog.txtö, which is erased each time you start up Revolution. The file is located inside the folder ôrevolution/components/save/ö. If you are experiencing problems, please send this file to support@runrev.com along with your bug report.

About what was new in version 1.1

See Also:

About what's new in version 2.1-2.1.1

This topic describes changes to Revolution between version 1.0 (July 2001) and 1.1 (November 2001).

The information in this topic is the same as the information contained in the "Read Me First" file that shipped with version 1.1.

Contents:

Overview of Version 1.1

New Features

Limitations and Known Issues

Overview of Version 1.1

Version 1.1 introduced changes and new features in the following major areas:

- ò File format change to accommodate new features
- ò Native compatibility with OS X
- ò Support for creating OS X applications
- ò Communication with SQL, Oracle, and ODBC databases
- ò HTTP support moved to Internet library
- ò FTP support added
- ò Support for Alpha Digital UNIX and SCO Open Desktop platforms removed

The engine version for Revolution 1.1 was 2.4 build 2.

New Transcript terms

aliasReference, alphaData, alternateLanguages, answer color, answer effect, armedIcon, backgroundBehavior, blendLevel, combine, create alias, dateFormat, delete URL, disabledIcon, extents, flip, imageData, imageSource, intersect, jpegQuality, keysDown, linkClicked, linkColor, linkHiliteColor, linkText, linkVisitedColor, longFilePath, maskData, matrixMultiply, median, merge, monthNames, mouseDownInBackdrop, mouseUpInBackdrop, paintCompression, popup command, QTEffects, record sound, recordFormats, recording, recordLoudness, resetAll, revdb_closecursor, revdb_columnbyname, revdb_columnbynumber, revdb_columncount, revdb_columnlengths, revdb_columnnames, revdb_commit, revdb_connect, revdb_connectionerr, revdb_connections, revdb_currentrecord, revdb_cursorconnection, revdb_cursorerr, revdb_cursors, revdb_disconnect, revdb_execute, revdb_isbof, revdb_iseof, revdb_movefirst, revdb_movelast, revdb_movenext, revdb_moveprev, revdb_query, revdb_recordcount, revdb_rollback, rotate, shortFilePath, showFocusBorder, specialFolderPath, split, standardDeviation, stop recording, transpose, union, useSystemDate, visited, visitedIcon, waitDepth, weekdayNames

Changed and extended Transcript terms

*, +, -, /, add, address, armed, average, backdrop, backgroundColor, backgroundPattern, convert, date, div, divide, do, each, effective, emacsKeyBindings, english, errorDialog, export, filename, files, folders,

hide, hiliteColor, hilitedIcon, htmlText, http, link, long, lookAndFeel, mod, multiply, platform, popup, rectangle, show, subtract, textStyle, time, underlineLinks, unlock screen, visual effect

Removed Transcript terms

executionError (supplanted by errorDialog)

scriptError (supplanted by scriptParsingError)

New Features

This section describes in detail all new features introduced in version 1.1.

Important! The file format of stack files changed between version 1.0 and 1.1. When you use 1.1 to open and save a stack file created in a previous version, Revolution automatically converts the file to the new format. You cannot use version 1.0 to open a stack file that has been saved in 1.1 or later. If you suspect you may want to backtrack to 1.0 for any reason - for example, if you have a license for 1.0 but not for 1.1 - be sure to make backup copies of your stacks before opening and saving them in 1.1.

OS X support

Revolution now runs natively on OS X, and you can use the Distribution Builder to create Carbon OS X applications. Revolution applications support the native Aqua appearance on OS X with the new `ôAppearance Managerö` setting for the lookAndFeel property.

Database access

Version 1.1 supports access to Oracle, MySQL and ODBC databases through the Database library, which you can include in standalone applications by checking the Database Libraries box in the Distribution Builder window. All platforms except OS X are supported (support for this platform will be added within a few weeks - as soon as driver issues on this platform are resolved).

The following new functions, which are documented in the Transcript

Dictionary, support database access:

revdb_closecursor, revdb_columnbyname, revdb_columnbynumber, revdb_columncount, revdb_columlengths, revdb_columnnames, revdb_commit, revdb_connect, revdb_connectionerr, revdb_connections, revdb_currentrecord, revdb_cursorconnection, revdb_cursorerr, revdb_cursors, revdb_disconnect, revdb_execute, revdb_isbof, revdb_iseof, revdb_movefirst, revdb_movelast, revdb_movenext, revdb_moveprev, revdb_query, revdb_recordcount, and revdb_rollback.

Also see the Database Manager in the Tools menu.

Professional license holders have full access to all RevolutionÆs database capabilities for ODBC, MySQL, and Oracle databases. Starter Kit users and Standard license holders have full access to ODBC databases using the database functions, but MySQL and Oracle database use is limited (for Starter Kit and Standard users) as follows:

ò Only one connection can be open at a time to an Oracle or MySQL database.

ò Oracle and MySQL connections are limited to a single database cursor.

ò A maximum of 20 rows can be retrieved from an Oracle or MySQL database.

ò A maximum of 64K of binary data can be retrieved from an Oracle or MySQL database.

The above limits do not apply to Professional license holders.

New file system features

Aliases (shortcuts, symbolic links) are now supported with the new `create alias` command and `aliasReference` function. Conversion between Windows long and short file names is provided by the new `longFilePath` and `shortFilePath` functions. You can find the pathnames to special folders such as the Preferences folder, Fonts folder, and other special folders on Windows and Mac OS using the new `specialFolderPath` function.

The new modifier `detailed` can be used with the `files` and `folders` functions, which return a list of extended filesystem information including file or folder size, access permissions, type signature and creator information, and creation, modification, access, and backup timestamps.

New date handling features

Revolution now supports dates and times in the user's preferred format and language. The `date` and `time` functions now accept the modifiers `english` and `system` to specify either the standard US format or the format according to the user's system preferences and language setting. The new `useSystemDate` property specifies whether the default is to use english or system dates and times. The `convert` command now accepts the `english` and `system` modifiers, as well as letting you choose a format to convert from.

The `date` function now supports the standard Internet date format used in email headers with the `internet` modifier.

The new `weekdayNames`, `monthNames`, and `dateFormat` functions let you parse and create your own date formats.

New and improved math functions and array handling

The `+`, `-`, `*`, `/`, `mod`, and `div` operators and the `add`, `subtract`, `multiply`, and `divide` commands now accept arrays as arguments. The `average` function now accepts arrays, as well as lists of numbers.

The new `standardDeviation` and `median` functions provide new statistical capabilities.

The new `transpose`, `matrixMultiply`, and `extents` functions can be used to manipulate arrays. The new `combine` and `split` commands convert arrays to delimited strings and back again. The new `union` and `intersect` commands compare and combine two arrays.

You can now use the new `repeat for each element` form of the `repeat` control structure to iterate over every element in an array.

New graphics and multimedia features

The alpha channel in imported PNG images is now fully supported, displaying pixel-by-pixel variations in transparency. The new `imageData`, `maskData`, and `alphaData` properties allow direct access to the binary data of an image. The new `paintCompression` and `jpegQuality` properties control the format of stored or exported images. The new `blendLevel` property can be used to make images partially or fully transparent.

The new `flip` and `rotate` commands change the orientation of an image or a portion of an image. Also see `ôFlipö` and `ôRotateö` in the Object menu.

The export command can now export any image without selecting it first, and can export image data to any container (such as a URL or variable). You can now export GIF images (with the purchase of a special license key—contact support@runrev.com for information).

Pictures (from Revolution image objects, files, or URLs) can now be embedded in text fields using the new `imageSource` property.

In addition to separate icons for their normal and highlighted state, buttons can now have a separate icon when they are disabled (`disabledIcon` property), have been clicked already during the current session (`visitedIcon` property), or when the mouse pointer moves into the button (`armedIcon` property).

The filename property of a player or image can now be a URL, allowing display of a movie or picture direct from a web site. Streaming media in players is now supported.

QuickTime special effects can now be used as visual effects by the `hide`, `show`, `visual effect`, and `unlock screen` commands. The old visual effects are still supported. The new `answer effect` command lets you select an effect and options for it. The new `QTEffects` function returns the names of currently installed QuickTime special effects you can use.

The player object no longer pauses when the user clicks in a playing movie.

The new `answer color` command displays the operating system's standard color-selection dialog box.

New linking capabilities

The `ôgroupö` text style has been renamed the `ôlinkö` text style, and the `showGroups` property has been renamed `underlineLinks` and now can be applied to individual stacks, as well as globally. You can continue to use the old term `ôgroupö` to set the text style.

The new `linkColor`, `linkHiliteColor`, and `linkVisitedColor` properties make grouped text (text whose style is `ôlinkö`) behave like Web browser links. These properties can be set globally or for individual stacks. You can assign hidden text to any grouped text using the new `linkText` property. Clicking grouped text sends the new `linkClicked` message, which includes the `linkText` of the clicked text. You can use these properties to fully emulate the behavior of links in a Web browser.

The new `visited` property of buttons and text groups reports whether the object has been clicked during the current session. Buttons which have been visited can display a separate icon using the new `visitedIcon` property. Grouped text which has been clicked is displayed in the `linkVisitedColor`.

Sound recording

The new `record sound` command records a sound file from the computer's sound input source. The available formats are returned by the new `recordFormats` function. The new `recording` and `recordLoudness` properties report whether sound is currently being recorded and the sound level. The new `stop recording` command stops the current recording.

FTP access, new Internet capabilities, and changes to HTTP access

All handling of HTTP URLs has been moved to the new Internet library, which you can include in standalone applications by checking the `ôInternet Librariesö` box in the Distribution Builder window.

The ability to use FTP URLs is now available. See the ftp keyword in the Transcript Dictionary for complete information on FTP uploading, downloading, and directory creation. Authentication is available by specifying a user name and password in the FTP URL. The new delete URL command removes a file or directory from an FTP server.

The new resetAll command closes all open sockets and stops all pending Internet transactions.

The filename property of a player or image can now be a URL, allowing display of a movie or picture direct from a web site. Streaming media in players is now supported.

AppleScript and OSA language support

On Mac OS and OS X systems, code written in other scripting languages (such as AppleScript and Frontier) is now supported with the do command. The new alternateLanguages function lists the available OSA languages.

New appearance and user interface features

The backdrop property can now display patterns as well as solid colors. Clicking the backdrop sends the new mouseDownInBackdrop and mouseUpInBackdrop messages.

The lookAndFeel property has a new setting, `Appearance Manager`, which is available on Mac OS and OS X systems. This setting uses the operating system's own routines wherever possible to draw user-interface elements such as scrollbars and buttons.

If the backgroundColor and backgroundImage properties of all objects in the object hierarchy are empty, the system background color or pattern is used.

The hiliteColor of an object can now be inherited from the object's owner.

A highlighted border around the active (focused) object is now supported with the showFocusBorder property. The effective modifier can be used with an object's rectangle property to determine its rectangle with or without the focus border.

Externals SDK

A Software Development Kit for producing external commands and functions to extend Revolution using C has been included.

Error handling changes

The executionError and scriptError properties have been removed. The information they reported is now sent as a parameter with the errorDialog or scriptParsingError messages, respectively.

New resource-handling functions

The new getResource function returns the binary data in a Mac OS resource. Its syntax is getResource(path,type,name). The name parameter can be either the name or the ID of the resource.

The new setResource function puts data into a resource. Its syntax is setResource(path,type,ID,name,flags,data). If either the ID or name parameter is empty, it will update an existing resource without changing its ID or name. The flags parameter can contain zero or more of the following characters in any order:

S (System heap)

U (pUrgeable)
L (Locked)
P (Protected)
R (pReload)
C (Changed)

The getResources function has been modified to include the flags parameter as described for setResources as its last parameter, and to return the values in the same order required for the setResource function (type, ID, name, flags).

Miscellaneous Transcript changes and additions

- ò The new waitDepth function returns the number of pending wait commands.
- ò The new backgroundColor property controls whether groups are automatically placed on new cards.
- ò The new keysDown function returns all the keys currently being pressed.
- ò The new merge function evaluates tagged expressions in a string and replaces the tagged expression with its value.
- ò The htmlText property now produces more standard HTML.
- ò Palette windows can now have a resize box.
- ò The popup command can now be used to display button menus as well as stack menus.
- ò The emacsKeyBindings property now uses the Control key on Mac OS systems.
- ò The address property no longer supports AppleTalk zones.

Miscellaneous other changes and additions

- ò Triple-clicking now selects the clicked line in fields.
- ò You can now drag and drop the selected text within a field.
- ò Support for working on a multiple-monitor system has been improved.
- ò You can now print entries from the Transcript Dictionary and the Encyclopedia.
- ò Importing HyperCard stacks has been made more robust.
- ò The script editor menus have been rearranged for better consistency.
- ò Alpha Digital UNIX and SCO Open DeskTop are no longer supported.

Limitations and Known Issues

Please report other bugs or issues to support@runrev.com.

ò Font, size, and style changes cannot be applied when the text selection is empty (a blinking insertion point) and no objects are selected. To apply a font, size, or style change to text, select the text, then choose the correct command from the Font menu.

ò Copying and pasting between programs supports only plain text, and styles are lost. (Copying and pasting text within Revolution preserves styles.) To work around this limitation, see the htmlText property in the Transcript Dictionary section of the documentation.

ò Selections in an image object cannot be copied (although you can copy and paste an image object as a whole within Revolution). This issue will be resolved in the near future.

ò Setting the explicitVars property to true will cause instability in the Revolution development environment. This issue is being actively investigated and will be addressed in a future version.

ò Scrollbar objects whose style is `scale` or `progress`, when they are oriented vertically, do not have the native appearance on Mac OS and Windows systems; vertical scale and progress bars always are drawn with the Motif appearance. (This limitation does not apply to scrollbar objects whose style is `scrollbar`, or to horizontal scale bars and progress bars, which are always drawn with the correct native appearance for the current platform. It also does not apply on Mac OS and OS X systems when the `lookAndFeel` property is set to `Appearance Manager`, which uses the standard system routines for display of user interface objects such as scrollbars.)

ò Recording a sound using the record sound command may produce unexpected results (such as a garbled sound file) if using the 22KHz (`better`) sound quality with some versions of QuickTime. This issue is being actively investigated.

ò Double-clicking a Revolution stack file to launch the Revolution application will open the stack without loading the development environment (which includes Revolution's menu bar). This behavior allows you to test your stack outside the development environment before you build a standalone application from it, but means you cannot edit stacks opened in this way because the development environment is not present. This issue is being actively investigated.

ò On Mac OS systems, the InputSprocket extensions have been reported to cause instability of some applications, including Revolution. If you experience instability and you do not require the sprocket extensions, try turning these extensions off in the Extensions Manager control panel.

ò Some Windows graphics drivers have compatibility problems with Revolution, in particular with images. The primary symptom is that images are drawn in the wrong colors. If you see this symptom, make sure you have the latest drivers from the vendor of your graphics card, and try turning off graphics acceleration (in the Performance tab in the System Control Panel). If this does not fix the problem, please report the bug to your graphics card vendor.

ò Revolution does not support running the Carbon (OS X) engine on Mac OS 8 or 9 due to Carbon library version compatibility problems.

ò The shell function and open process command are not yet supported when using the OS X engine. (They are supported using the Darwin engine.) The send to program command is also not yet supported, but you can use the do as appleScript form of the do command as a workaround.

ò When using QuickTime 5, the controller thumb in a player does not always update properly while a movie is playing. The workaround is to do something that will force the controller to be redrawn: for example, setting the player's `showController` property to false in a `preOpenCard` handler and back to true in an `openCard` handler, or the reverse with the `alwaysBuffer` property. This issue is being actively investigated.

ò When building a standalone on a Mac OS system, an extra file bearing the name of the standalone's main stack is left in the folder containing the final standalone application. You can delete this file when the build process is completed. We are looking into this issue.

A few cosmetic issues remain on OS X systems. These include the following:

ò Default buttons do not `throb`.

ò Sheet dialogs are not yet supported.

- ò Text is drawn with QuickDraw, so text may look slightly different in Revolution and Revolution applications.
- ò In modal dialog boxes, mouse clicks are blocked if a tool tip is visible.
- ò The interactive form of the import snapshot command does not display a "marching ants" rectangle.
- ò Menus whose menuMode is "pull-down" do not have the Aqua appearance. (This does not apply to menus displayed in the menu bar.)

About Revolution for HyperCard developers

See Also:

About porting HyperCard stacks, Getting Started Tutorial, Recipe for 99 Bottles of Beer on the Wall (using repeat), Recipe for 99 Bottles of Beer on the Wall (using send), dynamicPaths property, HCAddressing property, HCImportStat property, HCStack property

This topic is for new Revolution developers who have experience with HyperCard and the HyperTalk language, and who want to learn how to leverage their HyperCard knowledge and existing stacks to get a running start in Revolution.

Since Revolution can read HyperCard stacks, you can start out by converting or re-creating your existing stacks, then extend your work to include the new capabilities available in Revolution.

To fully understand this topic, you should know how to create stacks and write scripts using HyperCard. If you are an intermediate-level HyperCard developer, you have enough information to fully understand this topic.

Contents:

- From HyperCard to Revolution
- The Revolution Development Environment
- Transcript and HyperTalk
- Revolution Enhancements
- Common Traps for HyperCard Developers
- Tips for Good Revolution Habits
- WhatÆs Next?
- Summary

From HyperCard to Revolution

As an experienced HyperCard developer, youÆll find RevolutionÆs object model and development environment very familiar. You create objects in a stack window (which can have one or more cards), using the Objects menu and the Tools palette. You use the property inspector to change the appearance and behavior of these objects, and edit their code in the script editor. And you write handlers in the Transcript language, which is very similar to HyperTalk.

Revolution supports all the object types HyperCard does, plus more: stacks and cards, buttons, fields, images (bitmaps you can either paint in or display a file in), draw graphics, QuickTime players, and scrollbars. (The kinds of objects you can place on a card—fields, buttons, and so on—have the generic name of “controls”.)

There are a few basic conceptual differences to be aware of:

- ò You can have more than one stack in a file. Each stack is a separate window.
- ò You use groups instead of backgrounds to share objects between cards.

ò You create a menu bar by creating a special group of menu buttons, rather than by modifying HyperCardÆs own menu bar.

Stacks and windows

In HyperCard, each window is a stack, and each stack is a file.

In Revolution, each window is still a stack, but a stack file can contain multiple stacks. This means you can create a multi-window application in a single file. It also means that a stackÆs name can be different from the name of the file itÆs stored in. In fact, a stack can have a different name than what appears in its title bar. If the stackÆs label property is not empty, Revolution uses the label instead of the stack name in the title bar. You refer to the stack in scripts by its name property, just as in HyperCard.

Revolution gives you additional window options that werenÆt available to you in HyperCard. A stack can be displayed as a normal editable window, as a palette, or as a modal or modeless dialog box. This means you can easily create custom palettes and dialog boxes without using XCMDs, and include them in the same file as your main windows. A palette can contain any objects you want, instead of being limited to only buttons.

Stacks can have a resize box in the lower left corner, and can be resized either by the user or under script control. You can also set a minimum and maximum width and height. When the user resizes a stack, a `resizeStack` message is sent; handle this message if you need to re-position controls appropriately for the new window size.

Tip: Use the Geometry pane in the property inspector to have Revolution automatically move and resize objects in the stack when the user resizes the window.

Backgrounds and groups

In HyperCard, you place objects in the background layer, and each card in that background displays the background objects behind the card objects. Revolution does not use backgrounds in this way. Instead, it uses groups of objects, which can be displayed on a single card or on multiple cards that all share a group.

In Revolution, you can use the Group menu item (or the group command) to combine any set of selected objects into a group. Unlike HyperCard backgrounds, groups can be added to any card in the stack, and a card can contain more than one group. Groups can be interleaved with, or in front of, non-grouped objects, so youÆre not restricted to putting shared objects in the bottom layer and card-specific objects on top. Groups can be any size.

Groups are very flexible. You can select and move a group, show it, hide it, give it borders or scrollbars, and place it on any card or remove it from any card. Like a background, a group has a script, so you can script behaviors that apply only to the groupÆs objects. You can even create a group that contains other groups.

When you create a new card, groups on the current card can be automatically placed on the new card, just like the way HyperCard adds a new card to the current background. The groupÆs `backgroundBehavior` property controls this behavior.

However, it's important to remember that groups are not exactly the same as backgrounds. In HyperCard, every card belongs to a background. In Revolution, however, cards can exist without having any groups, and the groups belong to the cards they're on rather than vice versa.

Tip: To create a stack that behaves like a familiar HyperCard stack, you create a single group the size of the stack window, set its `backgroundBehavior` property to true, and choose Object menu Send to Back to make sure it is behind any card objects. This specific configuration is equivalent to a HyperCard background. (Of course, you can also branch out to using other capabilities of groups, such as placing multiple groups on a card.)

Menus and the menu bar

In HyperCard, you create popup menus by setting a button's `style` property to `popup` and placing the menu contents in the button. You change menus in the menu bar using the `create menu` command, and use the `doMenu` command to simulate choosing a menu item.

In Revolution, all menus (including menus in the menu bar) are implemented as buttons. Button menus are created in much the same way as HyperCard popup menus: You set the style property of the button to menu, and set its `menuMode` property to specify a pulldown menu, popup menu (called an `option menu` in Revolution), or combo box. These button menus can be placed anywhere in a stack.

To create a menu bar, you first create a button for each menu, setting the button's `menuMode` to `pulldown`. Then you combine those menus into a group, placed at the top of the stack window. (This is the normal location for menu bars on Unix and Windows systems.)

If you set the stack's `menubar` property to the name of the group, Revolution displays the group's menus as a normal Mac OS menu bar and changes the window size to hide the buttons. This means that the menu bar will appear in the proper place for each platform: on Mac OS and OS X systems, the menus appear at the top of the screen in the menu bar, while on Unix and Windows systems they appear at the top of the stack window, in accordance with user-interface standards for those platforms.

Tip: Use the Menu Builder feature to automatically create and place the buttons for a menu bar, set its appearance appropriately for any platform, and specify keyboard equivalents for its menu items.

The Revolution Development Environment

Revolution's development environment is similar to HyperCard's. There is a message box (the Revolution message box can execute multiple-line structures as well as one-line statements) and a Tools palette for working with objects.

To select an object of any type, you use the Pointer tool (arrow), instead of using a different tool for each object type. This means that you can select multiple objects of any type. For example, you can select both buttons and fields and move them all at once.

You access an object's properties by selecting the object, then choosing Object menu Object Inspector. The property inspector contains a menu at the top to let you show panes for basic properties (common to all objects), the object's custom properties, and properties specific to the object's type.

You open the script editor by selecting an object and choosing Development menu Object Script. The Revolution script editor accepts styled text, so you can use colors and fonts to emphasize portions of

your handlers, mark unfinished code, and so on. To search a script, choose Script menu Find and Replace.

Just as in HyperCard, you can test a handler as soon as it's written, and use your stack within the development environment. There is no compile step or separate runtime environment needed. However, if you want to test your stack in a "clean" environment without any of Revolution's own menus, palettes, or other parts of the development environment, you can use the "Suspend Revolution UI" item in the Development menu. When you're ready to build a standalone application, choose File menu Build Distribution. Here you can create standalone applications for any of Revolution's supported platforms.

The development environment's menus, palettes, dialog boxes, and all is built entirely in Revolution. This not only demonstrates the power of the Revolution engine, it means that as you advance, there is the possibility of exploring the Revolution user interface and customizing it for your needs.

Important! Double-clicking a stack in the Finder launches Revolution and opens the stack, but does not automatically load the development environment. You can use this capability to test how the stack operates without the development environment, before building it into a standalone application. If you accidentally launch a stack in this mode, close all the open windows to quit Revolution.

To override this launch behavior, hold down the Command key while double-clicking the stack, or else launch the Revolution application first and then double-click the stack to open it.

Transcript and HyperTalk

Revolution's scripting language is called Transcript. Since HyperTalk is one of Transcript's ancestors, almost all your knowledge of how to write HyperTalk code will translate easily.

As in HyperTalk, the script of an object is composed of one or more handlers that respond to messages sent to the object. All the familiar HyperTalk messages are supported (along with new ones), so you can set a button's script to read:

```
on mouseUp
    beep
end mouseUp
```

and it will react the way you expect it to, beeping once when clicked.

As in HyperCard, messages are sent first to the target object, then proceed along a message path containing all the owning objects. For example, if you click a button, the resulting mouseUp message is normally sent first to the button, then to the card the button is on, then to the stack the card is in.

Tip: In Revolution, built-in commands and functions do not traverse the message path, but are sent straight to Revolution. This means you cannot override the behavior of a built-in command or function by inserting a handler in the message path. (To create your own functions and commands, use a custom message handler.) However, this means Transcript handlers typically run from five times faster up to one hundred times faster than their HyperTalk equivalents. This speed increase means you'll find that many computing-intensive tasks that were too slow to be practical in HyperCard are feasible in Revolution.

You can refer to objects by name, number, or ID. As in HyperCard, the ID of objects never changes. (Stacks, which do not have IDs in HyperCard, and images, which don't exist as an object type in

HyperCard, are the exceptions. You can change a stack's or image's ID in Revolution.) No two objects in a stack can have the same ID.

Transcript supports all HyperTalk control structures (if...then...else...end if, repeat, pass, send, exit, return) as well as two new ones:

ò switch: A conditional structure (like if) for use when you want to check several possible cases.

ò try...catch...throw: A structure that executes a set of statements and catches any errors that occur during execution.

Transcript has approximately three times as many commands, properties, and functions as HyperTalk 2.4, so only a few of the new capabilities are discussed in this topic. Refer to the Transcript Dictionary in the Revolution documentation for complete language information.

Revolution Enhancements

Many of the features HyperCard developers have called for over the years are included in Revolution:

- ò fully native, script-controllable color for all objects
- ò easy cross-platform development for Windows and Unix
- ò vector (resizable) graphics
- ò true arrays
- ò easy handling of binary data and binary files
- ò effectively unlimited text in fields
- ò effectively unlimited script length

In general, limits on size are larger, script execution is much faster, more multimedia formats and object types are supported, and the number of built-in commands, functions, and properties is greater.

Revolution supports the HyperCard 1.x XCMD model, which means that most XCMDs and XFCNs that don't display a window will work in Revolution without modification for use on Mac OS systems. (If you want to use your externals on OS X, Windows, and Unix systems, they must be recompiled and usually rewritten.)

However, most HyperCard developers find that they rarely need to use externals because equivalents for most common externals are already part of Revolution. For example, you can use Transcript commands to:

- ò rename, move, and delete files;
- ò move resources among files;
- ò get the contents of a folder;
- ò set the cursor location;
- ò place a solid backdrop behind your stack;
- ò get URLs, post data to a web server, or communicate directly with other systems on the Internet via sockets.

Color is built in, and so is the ability to create and control palettes and dialog boxes. And the Transcript language is fast enough that many repetitive tasks that once required externals for speed become practical to implement as handlers instead.

Here are a few more enhancements and additions that you may find especially interesting:

Chunk operations

You can use all the familiar chunk types—characters, words, items, and lines—from HyperTalk in Transcript expressions.

Chunk expressions can include negative indexes to count backward from the end of a string. For example, word -2 of myString specifies the second-to-last word of the variable `myString`.

You can also loop over each chunk in a string:

```
repeat for each word currentWord in myString
  if currentWord is "George" then beep
end repeat
```

This form is much faster than repeat with `x = 1` to the number of words in `myString`.

The `is among` operator tests whether a specified chunk exists in a string:

```
if "Fred" is among the items of "Adam,Fred,George"
then doSomething
```

Transcript has `lineOffset`, `itemOffset`, and `wordOffset` functions in addition to HyperTalk's `offset` function, so you can find out which line, item, or word contains a specified string.

Unix-style regular expressions

In addition to chunk expressions, Revolution supports the powerful wildcard and regular-expression syntax via the `filter` command and the `matchChunk` and `matchText` functions. (Regular expressions and wildcard expressions are methods of specifying a text pattern you want to find.)

Direct URL addressing

Transcript treats URLs as containers, so you can work directly with the contents of any URL:

```
put URL "http://www.example.com/file.txt" into field "Text"
put myVar into URL "file:/Disk/Folder/File"
if URL "ftp://ftp.example.org/myfile/" contains "Help"...
```

Revolution supports standard file, http (web), and ftp URL types. It also adds `binfile` and `resfile` types for binary files and resource forks. You can put data into a URL or get data from a URL in the same way you would use a field, variable, or other container.

Modifying the message path

Revolution supports HyperTalk's `start using` command to let you use other stacks as script libraries.

You can also use the `insert script` command to place any object—not just a stack—into the message path:

```
insert script of button "Library" into back
insert script of card "Trap Keystrokes" into front
```

A script can be inserted either in back of the message path—after all objects—or in front of the message path—before any objects.

A script inserted into the front of the message path can handle a message before the target object itself receives the message. For example, if you place a mouseDown handler in an object's script and insert that script in the front, clicking a button sends the mouseDown message to that object first, and the button receives it only if the front script passes the message. Once you remove the script from the front using the remove script command, it no longer intercepts the message.

A script inserted into the back of the message path receives messages just before Revolution itself does, after the target object, its owning objects, and any stacks placed in use with the start using command.

Custom properties

A custom property is a property that you create and that is stored with the object, the same way built-in properties are stored. You can create custom properties for any object. These properties can contain any kind of data, of effectively unlimited length. You get and set them in the same way as built-in properties:

```
set the myCustomProperty of card "Intro" to true
get the hydrant of button ID 1045
put the puffball of me into myPuffball
```

HyperCard developers often use hidden fields to store data. But custom properties let you store data directly in any object instead, without any need for extra objects to hold text. Custom properties are also faster than fields. And unlike hidden fields, custom properties can hold binary data as well as text.

Inheritance of text appearance

You can set the textFont, textSize, and textStyle properties of a card, group, or stack (as well as their color properties). Any object whose text and color properties are not set will inherit the font, size, style, and color of the next object in the message path.

For example, if you set the textFont of a stack to Verdana, all the objects in the stack will automatically switch to Verdana (unless you've specified an independent textFont for them).

Binary data and nulls

In Transcript, you can read and write binary files which may contain null characters, ASCII zero. (In HyperTalk this is not possible because the null character cannot be handled by HyperTalk's data structures.)

The ability to read and write binary files and process binary data in variables allows you to create any type of file you wish, as long as you understand the format of the file type. For example, you can create MP3 files by writing the raw data directly to the file. You can also read and write binary data to a serial port—something that requires an external in HyperCard.

Associative arrays

Transcript supports array variables (variables that contain more than one value). The parts, or elements, of an array can be indexed by number or with any string.

Tip: Scan the Transcript Dictionary to explore more new language terms.

Common Traps for HyperCard Developers

Revolution has a high level of compatibility with HyperCard, and almost all your knowledge will translate easily. However, there are a few incompatibilities, and many areas where Revolution offers a different method of doing things. Here are the most common gotchas and some useful tips:

Special characters in scripts

A few special characters (that require the Option key) can cause problems in scripts, if you move the stack to a Windows or Unix system:

ò The HyperCard line-continuation character $\frac{1}{4}$ (option-L) will work in Revolution under Mac OS or OS X. However, it is not recognized under Windows or Unix, and a handler containing it will cause a script error. For cross-platform compatibility, use a backslash (instead. The backslash is recognized on all platforms.

ò Similarly, the not-equal character \neq , the less-than-or-equal character \leq , and the greater-than-or-equal character \geq can be used on Mac OS systems, but not on Windows or Unix. For cross-platform compatibility, use the synonyms \lt , \leq , and \geq .

HyperCard character:	Use instead:
$\frac{1}{4}$ (option-L, option-Return)	
\neq (option-=)	\lt
\leq (option-<)	\leq
\geq (option->)	\geq

Saving stacks

HyperCard saves changes to stacks automatically, but Revolution uses the more usual method of requiring the user to save changes explicitly. This has consequences in both the development environment and your applications.

Automatic saves:

In the development environment, your changes are not saved for you automatically. If you want changes saved automatically, one method is to write a plugin. Another is to simply include a closeCard handler with a save command in each stack you want to save automatically.

Saving data in standalone applications:

Revolution standalones cannot save themselves; they can only save separate files. (On Windows and Unix systems, a running application cannot modify itself. For consistency, Revolution also enforces this limitation on Mac OS systems.) This means that if the user enters data into your standalone and you want to save it, you'll need to provide a method, since data entered into any of the stacks in the standalone file itself will not be saved. There are two methods, and which one is best depends on what kind of data you want to save:

ò If the user needs to save a small amount of data, such as settings or preferences, it is often simplest to save this data in a text file in a quit handler. Your standalone's startup handler can then read the data from the file and use it to set up the standalone's window by placing text in fields, setting checkboxes, or whatever else is needed.

Tip: If you want to store your application's settings in the Preferences folder, use the `specialFolderPath` function to locate the Preferences folder.

ò If the user needs to save a larger amount of data—for example, in a database stack where the user is invited to make extensive notes—creating a stack file separate from your standalone is often the best method. This stack file serves as a document, owned by your standalone. Using this method, there is no need to move data into fields: when the separate stack file is opened, the data is already right there, ready to be modified and saved.

If users of your application need to save data, you will also need to provide this ability, since neither the Revolution development environment nor standalones created with it save stacks automatically. If you're creating your own custom menu bar, include a standard "Save" menu item in the File menu, or else make some other provision such as automatically saving windows when they're closed.

The Home stack

Like HyperCard, Revolution has a Home stack, whose script is part of the message path. In the Revolution development environment, the Home stack contains code used in the environment, and is not normally changed by users.

Installing utility handlers:

Since the Home stack is in the message path of all objects, many HyperCard developers place custom handlers in the Home stack script for easy availability while developing.

In Revolution, you accomplish the same thing by placing the handlers in an object's script, and use the `insert script` command to place the object in the message path as a backscript. An object inserted in the back of the message path receives all messages (unless they're intercepted by an earlier object in the message path) just before Revolution itself.

To automatically open such a utility stack whenever you launch Revolution, place the stack file in the "plugins" folder and choose Development menu Plugins Plugin Settings to open it automatically on startup. You can create as many plugin stacks as you want, and control when they load, how they're displayed, and which messages they receive.

Automatically searching for stacks:

Revolution's equivalent of the Search Paths card in the HyperCard Home stack is the `stackFiles` property. The `stackFiles` property consists of a list of stacks, by short name and by file path. If a handler refers to a stack that's not open, and the stack's name appears in the `stackFiles` property of the current stack, Revolution checks the stack file that's listed with it to find the stack.

Setting the userLevel:

Revolution has a `userLevel` property, but it is always set to 8 (to ensure that HyperTalk handlers that test the `userLevel` before doing an action will always conclude that the `userLevel` setting is high enough). Because the Revolution engine has no menus of its own, it's not necessary to block certain menu items from within your standalone by setting the `userLevel`; just don't include those menu items in your menu bar.

From resources to images

Since resources are a Mac OS-only service, for cross-platform compatibility Revolution stores cursors and icons as images instead of as resources.

You can import images from most common file formats using the import command (or by choosing File menu Import As Control Image File). You can use a color or black-and-white image of any size as a button icon. To keep these images out of the way, you can either hide them, or create a separate stack (which can be in the same file as your main stack) to hold them.

When you import an existing HyperCard stack, any ICON resources in the stack are imported as images and placed in a hidden group called `HC Icons`.

Overriding built-in commands and functions

In HyperCard, commands and functions pass through the message path, so a handler with the same name as a built-in command or function overrides its normal behavior.

In Revolution, built-in commands and functions are executed immediately rather than passing through the message path, so a handler with the same name as a built-in command or function will never be executed. If you need a command or function call to pass through the message path, use a custom command or custom function instead.

Everything in memory

When you open a stack file in Revolution, all its stacks are loaded into memory and kept resident until you purge the file. (Use the `destroyStack` property to remove files automatically from memory when you close them.) This makes for much faster access, since stacks are pre-loaded into memory and Revolution doesn't need to access the disk when you go to another card.

This also means that you cannot use stack files that are too large to fit into memory.

If you have a large collection of data in your stack, you may want to split it into multiple stack files. Or keep media, such as pictures, videos, and sounds, in external files and display these items in referenced controls. (Files in referenced controls are loaded into memory only when you go to the card the control is on.)

Dates, times, and the start of the eon

The Mac OS date routines (and therefore HyperCard) use midnight, January 1st, 1904 in the local time zone as the "beginning of time". The `seconds` function counts from this moment. However, since Revolution is cross-platform, it doesn't use the Mac OS-specific date. Instead, time begins at midnight, January 1st, 1970 GMT. This will affect your stacks if you store or manipulate the value of the `seconds` function.

The `ticks` function in Transcript goes back to the start of the eon, not to the last time the system started up.

The date and time functions and the `convert` command use the U.S. format for dates and times by default. (You can use the system keyword to specify that you want the date and time formatted according to the user's preferences.)

Language parsing

Transcript is pickier than HyperTalk about certain minor errors:

- The use of the word `the` before property names is more strictly enforced.

ò Literal strings must be enclosed in double quotes. (You can set the `explicitVariables` property to true to catch unquoted literals when you close the script.)

ò The `do` command and value function may require strings to be more strictly enclosed in double quotes, and values to be enclosed in parentheses.

ò You can't use a reserved word, such as a function, command, or property name, as a variable name or parameter name.

Pathnames

In HyperCard, file pathnames use the Mac OS convention of separating levels with a colon, and a file path might look like this:

Hard Disk:Folder:File

Revolution uses the Unix style for pathnames, which uses slashes instead of colons, on all platforms to simplify writing cross-platform handlers. The same file path, written Revolution-style, looks like this:

/Hard Disk/Folder/File

The folder that contains the current folder is indicated in a relative path by `../`. For example, if the current folder is `Folder 1`, the path to `Folder 2` on the same level is written `../Folder 2/`.

Revolution translates slashes to colons in pathnames, so if you have a file called `And/Or`, in a Revolution pathname it becomes `And:Or`. This avoids mistakes about whether the slash indicates a folder.

On OS X systems,

Tip: To see the path to a file, enter the following in the message box:

```
answer file "Choose a file: ";put it
```

This displays the file path for the file you choose. Examine the file paths of several files in different places to get an idea of what they look like.

Minor gotchas

Although almost all HyperTalk commands work in Transcript, there are minor syntax differences in a few of them. These may cause problems when using old HyperTalk code.

Visual effect quoting:

Transcript's visual effect command does not allow the visual effect name to be enclosed in quotes. The statement `visual effect "barn door"` does not work in Revolution, though it works in HyperCard.

Tip: Revolution can also use QuickTime-based visual effects, which opens up many new possibilities. For more information, see the visual effect command.

The `mark` property:

In HyperTalk, the property that specifies whether a card has been marked for further processing is called the `marked` property. In Transcript, this property is called the `mark`.

Stack location property:

In Revolution, a stack's location designates its center, not its top left corner. Also, to change a window's location or rectangle, you should set these properties of the stack, not the card.

Closing fields:

In Revolution, pressing the Enter key doesn't automatically close the current field. If you want to emulate this behavior, place an `enterInField` handler in the field's script (or the script of an object in the field's message path):

```
on enterInField
    select empty -- closes the field
end enterInField
```

Shared text fields:

In HyperCard, you can set a background field's `sharedText` property to true, put text into it, then set the property to false. HyperCard developers occasionally use this trick to store hidden text in a background field. In Revolution, fields whose `sharedText` property is false can't have shared text. Instead, use a custom property to hold the hidden text of the field.

Opening documents in another application:

In HyperTalk, you use the `open` command to open a document in an application. In Transcript, you instead use the `launch` command:

```
launch myDocument with myApplication
```

Music and the play command:

HyperCard's `play` command offers great flexibility in playing musical notes on sound samples stored in 'snd' resources. Revolution's play command is somewhat different:

- ò You can play sound files in any of the most common file formats.
- ò You can import a sound file into your stack, where it becomes an audio clip.
- ò You can record sound using the record sound command.
- ò You cannot play sampled sounds at different frequencies, so you cannot use the play command to specify musical notes.

The `itemDelimiter` property:

In HyperCard, the `itemDelimiter` is a global property, and if a handler sets it to a new value and then calls another handler, the delimiter in the second handler is also set to the new value.

In Revolution, the `itemDelimiter` is a local property, which means that changes to it affect only the handler in which it was changed. If a handler changes the `itemDelimiter` and then calls another handler, the `itemDelimiter` in the second handler has its original value, unless the second handler itself changes the property.

Referring to card and background objects:

As noted above, Revolution uses groups instead of backgrounds (although in some situations you can think of groups and backgrounds as being almost the same). Revolution does not assume either the card or background layer when referencing buttons and fields. In other words, you can specify `myButton` or `field ID 22` and Revolution will find the button or field either on the card or in a group, without needing to be told which.

The active stack window:

In most applications, the active window holds the current document, and menu commands operate on the active window. In Revolution, because you can open stacks as palettes and dialog boxes, this is not necessarily the case and occasionally the active window is not the frontmost stack. Use the `topStack` function and `defaultStack` property to control which stack is the active window.

Tips for Good Revolution Habits

While your stacks will work in Revolution without implementing any of the suggestions below, they will be more efficient and work better cross-platform if you keep these tips in mind as you start to work in Revolution.

Replace externals with built-in commands and functions

Many of the externals commonly used by HyperCard developers can be duplicated in Transcript code instead. XCMDs and XFCNs can be used only on Mac OS systems, and need to be recompiled (and in some cases rewritten) if you want to bring your work to OS X, Unix, or Windows. Even though you can use HyperCard 1.x externals in Revolution, replacing externals with a script whenever possible will simplify your development task and ensure that your stack is compatible with other platforms without any problems.

Here are a few examples of tasks that HyperCard developers commonly use externals for, and the equivalent Transcript terms:

Task:	Transcript term:
Copy files and folders	<code>revCopyFolder</code> command <code>revCopyFile</code> command
Move files and folders	<code>rename</code> command
Delete files and folders	<code>delete file</code> command <code>delete folder</code> commands
Work with aliases	<code>create alias</code> command <code>aliasReference</code> property
Get a folder's contents	<code>files</code> function <code>folders</code> function
Read HTML pages	<code>URL</code> keyword <code>htmlText</code> property
Upload via FTP	<code>ftp</code> keyword <code>libURLftpUpload</code> command
Download via FTP	<code>ftp</code> keyword <code>libURLftpDownloadToFile</code> command
Use binary files	<code>open file</code> command (binary option)
Use binary data via serial	<code>open file</code> command (binary option) <code>modem:</code> keyword <code>printer:</code> keyword <code>open driver</code> command

Work with resources	getResource function
	getResources function
	setResource function
	copyResource function
	deleteResource function
Display a custom dialog	modal command
Display a palette	palette command
Use QuickTime movies	player object

Displaying dialog boxes without externals:

Like HyperTalk, Transcript includes ask, answer, ask password, answer file, and ask file commands for displaying basic dialog boxes. To display a custom dialog box, you lay out the dialog box in a stack window, with whatever buttons, fields, images, or other objects you need. (Since modal dialog boxes don't have a close box, make sure to include at least one control whose script includes the close this stack command to dismiss the dialog box.) Then use the modal command to display the stack:

```
modal "My Dialog Stack"
```

Tip: If you display a modal dialog box but have forgotten to include a button to close it, use the shortcut to display a popup menu: Command-Control-Shift-click in the window, and choose "Top Level" from the "Stack Mode" submenu. This changes the stack back into a normal window.

How do you return the information about what the user chose back to the original stack that displayed the dialog box? There are many methods—for example, the modal dialog stack can set global variables or custom properties, which the originating stack can read. One method is to use the dialogData global property, which is especially designated for this purpose.

Suppose your dialog box displays a list field that lists all the cards in the stack, and you want to switch to the card the user clicked. Here's a script example:

```
-- This is the button that displays the dialog, in the
-- original stack:
on mouseUp
    -- first put the names into the list field in the
    -- dialog-box stack
    put the cardNames of this stack into

        field "List" of card 1 of stack "List Dialog"
    -- display the dialog box:
    modal "List Dialog"
    -- The handler halts here until the dialog box is
    -- dismissed. Then it continues to the next line:
    go card (the dialogData)
    -- The mouseDown handler in the dialog box already
    -- the dialogData property to the card name the user
    -- clicked, so we can just go to that card.
end mouseUp

-- This is the list field, in the dialog-box stack:
```

```

on mouseDown
-- The user has clicked a line in the list, so
-- we set the dialogData (which the mouseUp handler
-- in the button will use:
set the dialogData to (the value of the clickLine)
-- now we've set the dialogData property, it's time
-- to get rid of this dialog box:
close this stack
end mouseDown

```

Replace hidden fields with custom properties

Occasionally, you'll need to store data (such as a card modification date) so that it's not visible to users, but scripts can access it. In HyperCard, it is common practice to store such data in a hidden field on the card:

```

on closeCard -- HyperTalk-friendly example
  put the seconds into field "Hidden Date"
end closeCard

```

This handler will also work fine in Revolution. However, a better way is available. In Revolution, you can set a custom property instead. Custom properties can hold any data you want to place in them, are saved with the stack, can be associated with any object, and are used in handlers just like built-in properties.:

```

on closeCard -- Transcript-friendly equivalent
-- creates a custom property called "lastAccessDate"
set the lastAccessDate of this card to the seconds
end closeCard

```

You can refer to this custom property in exactly the same way you refer to other properties of the card.

Tip: To see and edit an object's custom properties, select the object, choose Object menu Object Inspector, and choose "Custom Properties" from the menu at the top of the property inspector.

Replace paint graphics with graphic objects

HyperCard provides only bitmapped image tools. You create card and background pictures using the paint tools built into HyperCard. Revolution also lets you create images, which are objects that contain paint bitmaps. Image objects are similar to the card and background pictures in HyperCard, but they can be any size, you can include any number of them on a card or in a group, and of course they can use color. Images can also have scripts of their own.

Revolution also provides graphics, which are resizable shapes. It's easier and more efficient to use graphic objects instead of paint images for such things as lines and boxes: you can resize them without distortion, and they generally take less memory and disk space than equivalent image objects. You should use images only for complex pictures (such as photos) that go beyond simple shapes.

You can create graphics by using the tools in the Tools palette, by choosing Object menu New Control, or by using the create command in a handler. A graphic's style property determines its shape

(rectangle, oval, and so on). Graphic objects can be easily moved, resized, and changed, either by setting the object's properties in a handler or by using the property inspector.

Replace idle handlers with delayed sends

At times you will need to execute one or more commands periodically. In HyperCard, it is common practice to place such commands in an idle, mouseStillDown, or mouseWithin handler.

While Revolution supports idle, mouseStillDown, and mouseWithin, it's more efficient to use the send command to send a message at a specified future time. This means you can place the commands to be executed in a handler, and at the end of that handler, send a message to execute it again after the delay you specify.

For example, suppose you want to annoy the user by beeping every ten seconds. In HyperTalk, you might use this idle handler, which uses a global variable to keep track of the next beep time:

```
on idle -- HyperTalk example
  global nextBeepTime
  if the seconds > nextBeepTime then
    beep
    put the seconds + 10 into nextBeepTime
  end if
end idle
```

In Revolution, the send command has an additional option that lets you delay before sending the message. With this capability, you can execute a handler, and within that handler, send a message to execute the same handler again after ten seconds. This example annoys the user similarly to the idle handler above, using the send command instead:

```
on annoyUser -- Transcript-friendly equivalent
  beep
  send "annoyUser" to me in 1 second
end annoyUser

on openCard -- starts the chain of messages going
  send "annoyUser" to me in 1 second
end openCard
```

Once the first "annoyUser" message is sent, the send statement inside the handler automatically queues the next one.

This approach offers greater efficiency, since Revolution does not need to continually execute an if...then statement in an idle handler to determine whether enough time has passed. It also gives you much greater control over timing. Use of the idle handler to perform lengthy tasks may cause Unix systems to lock up, and this alternative technique avoids such potential problems.

Tip: To cancel a pending message that has been sent with a time delay, use the cancel command.

Use mouseMove instead of polling the mouse state

When you need to monitor the mouse's position, it is usual in HyperCard to poll the mouse position within a repeat loop (or in an idle or mouseWithin handler):

```
on mouseUp -- HyperTalk example
  repeat while the mouseLoc is within the rect of me
    put the mouseLoc into field "Mouse Position"
  end repeat
end mouseUp
```

As with the idle and mouseWithin messages, this kind of construction is supported by Revolution. However, it requires Revolution to check the mouse position continually, even when it hasn't changed. It's more efficient to use a mouseMove handler in an object's script to track the mouse's motions while the pointer is within the object.

```
on mouseMove -- Transcript-friendly equivalent
  put the mouseLoc into field "Mouse Position"
end mouseMove
```

Use URLs instead of open file

You can use URLs as containers, and use commands like get and put with them. This applies to URLs of all types: ftp and http URLs (which are used to upload and download files), as well as file, binfile, and resfile URLs, which can be used to put data into or get data from local files.

For example, to read a file's contents into a field, you might use this set of HyperTalk statements:

```
open file "Disk:Folder:File" -- HyperTalk example
read from file "Disk:Folder:File"
put it into field "File Contents"
close file "Disk:Folder:File"
```

You can use a similar sequence of statements in Revolution (after changing the pathname to use slashes instead of colons), but you can also use this simple statement to replace all four lines:

```
put URL "file:/Disk/Folder/File" into field "File Contents"
```

What's Next?

Learning More About Revolution

As an experienced HyperCard developer, your next step is to familiarize yourself with the new user interface and the custom libraries.

Recommended tutorials

The first tutorial, "Getting Started", takes you through the process of creating and building a basic application. Although you will already be familiar with most of the concepts discussed in this tutorial, it will be helpful to invest an hour or less to go through the steps and create the application. Doing so will familiarize you with the different menu items and tools used to create stacks, and make it easier to locate them in the future.

You should also make time to go through the second tutorial, "The Menu Builder". The process of creating menus is one of the areas where Revolution differs the most from HyperCard, so a little time

spent on this subject will help avoid confusion in the future. You may also want to read through the topic "About menus and the menu bar" to obtain the conceptual background underlying the Menu Builder.

The other tutorials will also be useful to you in learning about special Revolution features. In particular, the final tutorial, "Independent Study", includes a full Revolution application with heavily-commented code.

Porting HyperCard stacks

One of the first things you want to do with Revolution may be to convert your HyperCard stacks to Revolution stacks. Revolution reads HyperCard stacks directly—you simply open the stack in Revolution—but some conversions and changes may be needed, depending on how complex the HyperCard stack is. The topic "About porting hyperCard stacks" describes the process in detail.

Jacqueline Landman Gay of HyperActive Software has written an extensive tutorial describing the conversion of a typical HyperCard application to Revolution. This tutorial is available on the web at <http://www.hyperactivesw.com/mctutorial/rrtutorialtoc.html>, and contains many tips and tricks to better adapt your existing stacks and HyperTalk code to use in Revolution.

Learning from the Cookbook

The Transcript Cookbook is a section of the documentation that contains annotated code "recipes" for doing various tasks.

As an experienced HyperTalk programmer, you can look through the Cookbook to see Transcript code in action and get a better feel for the differences between good HyperTalk practice and good Transcript practice. You'll find that some of the recipes are written the same way you would write them in HyperTalk, while others illustrate different methods and new capabilities. For example, the topic "Recipe for a card slideshow" demonstrates the use of delayed sends for a task that you would probably use "idle" for in HyperTalk.

To see the Cookbook, click "Transcript Cookbook" in the main Documentation window. Or choose any topic starting with "Recipe for..." from the Development Guide section of the documentation, or from the See Also menu in any documentation topic.

Summary

In this topic, you have learned that:

- ò Revolution's object model and development environment are similar to HyperCard's. Its language, Transcript, is similar to HyperTalk, though larger with more commands, functions, and properties.
- ò You can have one or more stacks in a file. Each stack can be displayed as a window, modeless dialog box, modal dialog box, or palette.
- ò A group is a set of objects that have been combined into one. The group has its own script, and so does each of its objects. Groups can be placed on a single card or shared between multiple cards.
- ò All menus in Revolution are created from buttons; a menu bar is created from a group of buttons. The Menu Builder automates the task of creating a menu bar.

ò You can use HyperCard 1.x XCMDs and XFCNs in Revolution. However, in most cases, externals are not needed because the functionality of the most common HyperCard externals is already built into Revolution: color, QuickTime playback, file handling, URL and Internet access, and so on.

ò Most of your HyperTalk code will go on working in Revolution, but for certain constructs, Revolution offers a better or more efficient way of writing the same functionality. Look through the Transcript Cookbook to find examples of Transcript code.

About porting HyperCard stacks

See Also:

About Revolution for HyperCard developers, About Colors and Color References, About Menus and the Menu Bar, About Printing, Getting Started Tutorial, Menu Builder Tutorial, Why can't Revolution find a handler?, dynamicPaths property, HCAddressing property, HCImportStat property, HCStack property

This topic is for Revolution developers with some experience who need to import existing HyperCard stacks into Revolution, and want a guide to the best way to accomplish a hassle-free conversion process.

Since Revolution can read HyperCard stacks, you can start out by importing or re-creating your existing stacks, then extend your work to include the new capabilities available in Revolution.

To fully understand this topic, you should be an experienced HyperCard developer, and should know how to create objects and write short scripts in Revolution. If you have gone through the "Getting Started" tutorial and have read "About Revolution for HyperCard Developers", you have enough information to fully understand this topic.

Contents:

- Differences Between HyperCard and Revolution

- Preparing to Import

- The Import Process

- Rewriting for Revolution

- What's Next?

- Summary

Differences Between HyperCard and Revolution

Revolution can read HyperCard stacks directly and convert them to Revolution stacks. However, some HyperTalk features are implemented differently in Revolution, and if your stack uses any of these features, you may need to make some changes to make it work properly.

The most common areas that need to be addressed are:

Color:

Color is built into Revolution, and Revolution does not support color added to HyperCard stacks using the AddColor XCMD architecture. If you have colorized your HyperCard stack, you will need to re-color it after importing, using Revolution's color properties.

Resources:

Revolution uses objects, rather than resources, for such items as sounds and icons. This ensures that a stack can be easily moved to Unix and Windows systems, since these systems do not support a Mac-style resource fork. Sounds, icons, and cursors (resources of type 'snd ', 'ICON', and 'CURS' respectively) are automatically converted to Revolution objects when the stack is imported, but all other resources in the stack are discarded. If your stack requires other resources, you will need to move them into the new stack.

Menu bars:

Unlike HyperCard, Revolution implements menus as objects. There is no built-in menu bar, as there is in HyperCard standalones; if you don't create a menu bar for a standalone, it will have a blank menu bar when you run it. Because menus are handled differently in Revolution, almost all HyperTalk code that affects the menu bar must be rewritten.

Development environment:

Because Revolution's development environment is so different from HyperCard's, most scripts that alter the environment, such as showing and hiding palettes, do not work without changes. An exception is the message box: you can use the same syntax as in HyperTalk to show, hide, and move the message box.

Report printing:

Revolution does not include HyperCard's Print Report engine, so code that prints reports must be rewritten using Revolution techniques.

Preparing to Import

The first time you open a HyperCard stack in Revolution, it converts it to Revolution and attempts to compile all the scripts in the stack. This means that any scripts that contain HyperTalk code that's not compatible with Transcript will cause script errors when you import the stack.

To avoid excessive error reports, make the original HyperCard stack as Transcript-compatible as possible before opening it in Revolution. It's easiest to rewrite or comment out lines of HyperTalk code that may cause problems. Then after converting the stack to a Revolution stack, you can test the possible problem areas one at a time, and rewrite them if necessary.

Note: All these changes can be made instead after opening the stack in Revolution. But if you have access to a Macintosh and a copy of HyperCard, it's easier to do the preparation beforehand, since it avoids having to wade through script errors.

1: Alter the scripts

The first step is to open the stack in HyperCard and make changes in the scripts, to avoid error messages when the stack is imported.

Comment out or completely remove all lines of scripts that contain these HyperTalk terms:

- domenu
- create menu
- delete menu
- menuMessage
- addColor
- open report printing
- soundchannel
- calls to XCMDs and XFCNs

In general, handlers that refer to menus, report printing, or XCMDs and XFCNs should be commented out. After you open the stack in Revolution, you can refer to the commented-out portions when writing Transcript code to replace them.

Tip: The easiest way to manage repetitive changes in a HyperCard script is to use the script editor's search and replace commands. For stack-wide searches, use the built-in `searchscript` command from the message box. For example, to find every instance of the command `doMenu` in a stack's scripts, type this into HyperCard's message box:

```
searchscript "doMenu"
```

HyperCard will open the first script that contains the word `doMenu`. Once the script editor is open, type Command-G to locate each instance and change it, or use the script editor's search and replace feature.

Color:

You can completely remove code that manages color, because color in Revolution is handled by setting object properties and does not rely on scripts. (If some of your scripts change the color dynamically—for example, changing the color of a button when the mouse moves over it—comment out the handler instead of removing it, so you can use it for reference later when you rewrite the handler in Transcript.)

Sound:

Transcript's play command does not support the musical notation options of the HyperTalk command. If your scripts simply use the play command to play a sound resource, no change is needed. Only notated music must be replaced. (If you need to play a song, you can substitute a pre-recorded sound for HyperCard's notation syntax.)

Language differences:

Revolution is pickier than HyperCard about a few types of minor errors. In particular:

- ò Revolution does not allow reserved words (such as the names of functions, commands, or properties) to be used as variable names or parameter names. Variable names such as `date` or `name` will cause errors.

- ò Unquoted literal strings will usually cause errors. You should enclose all literal strings in double quotes.

- ò Revolution does not allow surrounding visual effect names with double quotes. If your `visual effect` commands include quotes, remove them before importing:

```
visual effect "zoom out" -- remove the quotes first  
visual effect zoom out -- ready for import
```

2: Move HyperCard icons and sounds into the stack

If your stack uses any of HyperCard's built-in icons, any icons you have stored in your Home stack, or any of the built-in sounds (harpsichord, boing, flute, dial tones), use ResEdit or the Resource Mover in the Power Tools stack to copy those resources into your stack.

Revolution will import sounds and icons, but cannot see them if they are not in the stack being imported. Moving them now will save you some work later.

3: Compact the stack

Choose File menu Compact Stack to compact the stack. This is an important step that cleans up the stack data and prevents problems when importing the stack into Revolution.

Preparing non-standard stacks

You may have some stacks you need to import that present special problems. For example, you may have a standalone application but not its source stack. Or your stack may have been password-protected so you can't change its scripts. Or you may have a HyperCard stack you need to convert, but lack a copy of HyperCard or a Macintosh to run it on. Such stacks can be imported into Revolution, but require some special preparation first.

If you have the standalone but not the original stack:

It is always preferable to work with the source stack, but if you do not have access to the original stack, a standalone HyperCard application can be converted back to its stack form if you have access to a Macintosh. The process of building an application from a stack embeds all the resources HyperCard needs into the stack, and these generic HyperCard resources need to be removed, while leaving the resources that are unique to this stack in place.

Important! Always work on a copy of the standalone application, since mistakes will make it unusable.

To convert a HyperCard standalone back to a HyperCard stack, follow these steps:

1. Open the standalone with a resource editor such as ResEdit.
2. Delete all resources of type 'XCMD', 'XFCN', 'xcmd', and 'xfcn' that your stack doesn't use. (During conversion, Revolution ignores these resources, but if your stack relies on an XCMD or XFCN then you should leave it in the stack to avoid script errors when you open the stack in HyperCard.)
3. Delete all other resources except those of type 'snd ', 'ICON', and 'CURS'.
4. Open each of the above three resource types and delete any sounds, icons, or cursors that the application does not use.

When making a standalone, HyperCard adds all its built-in icons by default, so usually you can remove most or all of the 'ICON' resources. If the only 'CURS' resource you see is the browse hand, the entire 'CURS' resource type can be deleted since Revolution has its own browse hand.

5. Choose File menu Get Info For application name. In the dialog box that appears, change the Type to `ôSTAKö` and the Creator to `ôWILDö`. (Make sure to capitalize both.)
6. Save the file and quit ResEdit.

The file will now be a plain HyperCard stack, which you can edit in HyperCard and then open in Revolution.

If the stack is password-protected:

There are two types of protected HyperCard stacks: those with passwords that disallow access to certain menus and features, and those with passwords that prevent the stack from being opened at all.

The second type of protected stack cannot be accessed without the original password. The first type, however, can be stripped of its password using an XCMD called Unprotect. (Unprotect is available online from the Files section of the HyperCard Mailing List.)

If you don't have HyperCard:

If you have a stack but not a copy of HyperCard, you can open the stack directly in Revolution. Without the preparation steps discussed above, expect more error messages when Revolution converts the stack.

If you don't have a Macintosh:

You cannot use the HyperCard application on a Unix or Windows system. However, if a HyperCard stack file is accessible to such a system (for example, via the Internet), you can download the stack and open it in Revolution. Stacks can be imported in raw binary, BinHex, and MacBinary formats. The latter two preserve resource fork information including icons, cursors, and sounds.

Without a Macintosh or a copy of HyperCard, you will be unable to prepare the stack for Revolution, so expect more error messages when Revolution converts it.

The Import Process

To import a HyperCard stack, you simply choose File menu Open Stack. Revolution reads the file directly and imports it. The next time you save the file, it will be saved as a Revolution stack.

Tip: When importing a HyperCard stack, Revolution sets the stack's HCStack property to true. Check this property if you need to know whether a stack originated in HyperCard.

Understanding error messages

Even after preparing the stack in HyperCard, there are likely to be some remaining script errors. There are two general types of errors: compile errors and execution errors. Compile errors are generated when Revolution encounters terms or syntax that it does not understand. Execution errors are generated when the syntax is correct, but Revolution cannot run the handler for some other reason.

Important! If a handler contains a compile error, none of the handlers in the same script can be compiled. This means that if one handler in a script has a compile error, none of the other handlers in that script will work until the error is corrected. This may cause execution errors in other scripts, if those scripts depend on handlers in the script that has the compile error: if a script won't compile, Revolution will not be able to find any of its handlers if they are called by a handler in a different script.

Both types of errors are listed in the Error window. To locate errors, go to each card in the stack, and if errors occur, click the Script button in the Error window to view the portion of the script that is causing the problem. The information in the Error dialog will also give you a clue as to what went wrong.

Most post-import errors are due to one of the following problems:

- ò Literal strings do not have double quotes around them. (HyperTalk is more permissive than Transcript about quoting literal strings, and unquoted strings will work in some situations in HyperCard but not in Revolution.)

- ò A reserved word (that is, a word that is part of the Transcript language) has been used as a variable name or parameter name. Change these to a different unique name.

- ò The word `the` has been omitted from a property reference. Transcript is more particular about this than HyperTalk. References such as `name of this stack` should be changed to `the name of this stack`.

- ò The name of an effect in a visual effect command is quoted. Remove all quotes: Transcript does not expect them.

ò The script uses a HyperTalk term that should be replaced by the equivalent Transcript term. For more information about HyperTalk terms and Transcript equivalents, see "Common Traps for HyperCard Developers" in the topic "Revolution for HyperCard developers".

Tip: If you need to make repeated changes throughout a stack's script, choose Edit menu Find and Replace, and choose the "Script" option in the Properties section of the dialog box.

Where stack objects are placed

For the most part, all objects and scripts remain where they were in the HyperCard stack. There are three exceptions:

ò Backgrounds are converted to groups, and each group is placed on the appropriate card or cards. The original background scripts become the scripts of these groups.

Caution! Be careful about using the Ungroup command. A group's script remains only as long as the group exists. If you ungroup it, its script is lost.

ò Icons and cursors are imported as images. All imported icons are placed in a group called "HC Icons", which is not placed on any card. If you plan to replace all of the icons with color substitutes, you can delete this group entirely when you no longer need the icons for reference. Imported cursors are placed in a group called "HC cursors".

ò Sounds are imported as audio clips.

Icons are automatically assigned to the correct buttons, and sounds and cursors will work when called in handlers.

Rewriting for Revolution

To smooth out differences in syntax, Revolution automatically sets two stack properties when you import a HyperCard stack. An imported stack's HCAddressing property is automatically set to true, which causes Revolution to assume that references to buttons refer to card buttons and references to fields refer to grouped (background) fields. The dynamicPaths property is also set to true, which implements HyperCard's dynamic path in case your stack depends on that behavior.

If the original HyperCard stack used no color and contained no externals or custom menus, very little will need to be rewritten. More elaborate stacks need some additional clean-up work.

Menus

If your stack uses a custom menu bar, re-create it. The easiest way to create a new menu bar is to use the Menu Builder. (To open the Menu Builder, choose Tools menu Menu Builder.)

Color

All color objects should be recolored using Revolution's native color properties. To color an object in Revolution, you select it and choose Object menu Object Inspector, then choose "Colors & Patterns" from the menu at the top of the property inspector.

You can also use a handler to set the eight color properties and eight pattern properties of each object. This example handler makes the text of all the buttons on the current card orange:

```

on mouseUp
  repeat with thisButton = 1 to the number of buttons
    set the foregroundColor of button thisButton to "orange"
  end repeat
end mouseUp

```

Note: In Revolution, colors are inherited by objects in the object hierarchy. For example, if you set the foregroundColor property of a card, all objects on that card inherit the card's foregroundColor, if the object's own foregroundColor is not set.

For more information about color, see the topic "About Colors and Color References".

Sounds

Most imported HyperCard sounds will play correctly without alteration. In rare cases, these sounds may need to be resampled at a different rate or saved in a different format to be compatible.

Tip: The most universal format for cross-platform deployment is AU with 2:1 mu-law compression.

Special characters in scripts

Certain characters created with the Option key will work in scripts when run on a Macintosh, but will cause script errors if you move your stack to a Unix or Windows system. To ensure your stack is cross-platform ready, use the following synonyms:

HyperCard character:	Use instead:
¼ (option-L, option-Return)	
_ (option-=)	<>
_ (option-<)	<=
_ (option->)	>=

(During the import process, Revolution automatically converts the option-L character to the backslash)

Pathnames

Revolution uses Unix-style file paths, with slashes (/) instead of colons (:) separating the levels between folders. If your scripts contain hard-coded file paths, you must replace them with Revolution-style paths.

HyperCard file path:	Becomes:
Hard Disk:Folder:	/Hard Disk/Folder/
Hard Disk:Folder:File	/Hard Disk/Folder/File
::Parent Folder:Subfolder:File	../Parent Folder/Subfolder/File
Filename/With/Slash	Filename:With:Slash

For more information about file paths, see the topic "About filename specifications and file paths".

Speech

Transcript does not include a text-to-speech command. However, a free external for speech on Mac OS and Windows is available on the Runtime Revolution web site at

<<http://www.runrev.com/revolution/developers/developerdownloads/externalscollection.html>>

XCMDs and XFCNs

Revolution supports the HyperCard 1.x XCMD model, which means that most XCMDs and XFCNs that don't display a window will work in Revolution without modification for use on Mac OS systems. (If you want to use your externals on OS X, Windows, and Unix systems, they must be recompiled and usually rewritten.)

However, almost all HyperCard externals can be successfully replaced using built-in Revolution features. Some common replacement techniques include:

Task:	Transcript term:
Copy files and folders	revCopyFolder command revCopyFile command
Move files and folders	rename command
Delete files and folders	delete file command delete folder commands
Work with aliases	create alias command aliasReference property
Get a folder's contents	files function folders function
Read HTML pages	URL keyword htmlText property
Upload via FTP	ftp keyword libURLftpUpload command
Download via FTP	ftp keyword libURLftpDownloadToFile command
Use binary files	open file command (binary option)
Use binary data via serial	open file command (binary option) modem: keyword printer: keyword open driver command
Work with resources	getResource function getResources function setResource function copyResource function deleteResource function
Display a custom dialog	modal command
Display a palette	palette command
Use QuickTime movies	player object

HyperCard's built-in XCMDs include the `flash`, `picture`, and `palette` commands:

ò The functionality of the `flash` command can be emulated by changing the ink property of objects to change their appearance.

ò Revolution includes an image object that can display either an internal image or a picture file, similar to the `picture` command. PICT resources in a stack are not converted automatically when the stack is imported, but you can import images manually and place them on cards, or display them from a file.

Tip: For best cross-platform compatibility, use JPEG, GIF, or PNG formats for pictures.

ò You can make any stack into a palette in Revolution, and palettes can contain any type of object. To display a stack as a palette, use the palette command:

```
palette stack "My Palette"
```

Externals that are commonly used in HyperCard to speed up long script processes are almost never necessary due to Revolution's native speed. These externals can be rewritten as ordinary handlers.

The `userLevel` property

Changing the `userLevel` will not cause an error in Revolution, but it has no effect: there are no separate user levels in Revolution, since all menus are created by the stack author and all functionality is specifically scripted. If you want to forbid access to certain features or menus, simply do not include them in the stack. The `userLevel` property always reports 0 in Revolution.

Overriding built-in commands and functions

In Revolution, built-in commands and functions are executed immediately rather than passing through the message path, so a handler with the same name as a built-in command or function will never be executed. You will need to change such handlers to use a different name, and rewrite handlers that call the command or function to use the new name.

Saving

Like most applications, Revolution saves its files only at the user's request. Remember to save your work frequently during development. If your stack accepts user input, remember that users will need to save their work as well, so be sure to include a "Save" menu item in the File menu. This script in a menu will save the current stack:

```
on menuPick theItem -- triggered when user chooses an item
  switch theItem
  case "Save" -- user chose the "Save" menu item
    save this stack
    break
  end switch
end menuPick
```

To emulate HyperCard's auto-save feature, place this handler in the stack script:

```
on closeCard
  save this stack
end closecard
```

If you are planning to distribute your stack as a standalone application, the process is a little different. On Windows and Unix systems, a running application cannot modify itself. For consistency, Revolution also enforces this limitation on Mac OS systems. Stacks that are not compiled as applications can save their data on any platform.

Printing

Revolution supports HyperCard's `print card` command (as well as the form `print card from point to point`) and batch printing with the `open printing` and `close printing` commands. To print plain text or the contents of a field, use the `revPrintText` and `revPrintField` commands.

Revolution does not include HyperCard's report printing features, so scripts that rely on report printing must be rewritten. In general, to create a report in Revolution, you create a stack the size of the printed page, place text fields and other objects on it, and print the card. If the fields contain more text than will fit on a single page, the fields can be scrolled and the card reprinted multiple times until all the text has been printed. Use the `pageHeights` property to determine how much to scroll long fields in this case. When printing on a Windows system, set the `formatForPrinting` property of the stack to `true` in order to get an accurate scroll measurement based on Windows printing font metrics.

What's Next?

Learning the Revolution Way

As you continue developing in Revolution, you'll learn tricks and new commands that will let you make your existing stacks more efficient and full-featured. To get started in this process, review `ôTips for Good Revolution HabitsÆ` in the topic `ôAbout Revolution for HyperCard developersö`. You can also scan the Transcript Dictionary to learn more about Revolution's capabilities, and the Transcript Cookbook to find code examples.

Jacqueline Landman Gay of HyperActive Software has written an extensive tutorial describing the conversion of a typical HyperCard application to Revolution. This tutorial is available on the web at [<http://www.hyperactivesw.com/mctutorial/rrtutorialtoc.html>](http://www.hyperactivesw.com/mctutorial/rrtutorialtoc.html), and contains many tips and tricks to better adapt your stacks to Revolution.

Summary

In this topic, you have learned that:

- ò When porting a HyperCard stack to Revolution, the areas most likely to need some changes are color, custom menu bars, and report printing.
- ò The conversion process will be easier if you open the stack in HyperCard first and comment out or remove potentially problematic code.
- ò To import a HyperCard stack, you simply open the stack in Revolution. The stack is automatically converted, and sound, cursor, and icon resources are moved into it.
- ò There are some additional syntax differences that may require some testing and rewriting of scripts after the conversion process is complete. Revolution assists this process by setting the `HCAAddressing` and `dynamicPaths` properties of newly-converted stacks to `true`, eliminating some of the possible problems.

About Revolution for MetaCard developers

See Also:

Getting Started Tutorial, Independent Study Tutorial, How to convert a MetaCard stack to Revolution, buildNumber function, version function

This topic is for new Revolution developers who have experience with MetaCard and the MetaTalk language, and who want to learn how to leverage their MetaCard knowledge and existing stacks to get a running start in Revolution.

To fully understand this topic, you should know how to create applications and write scripts using MetaCard. If you are an intermediate-level MetaCard developer, you have enough information to fully understand this topic.

Contents:

From MetaCard to Revolution

Importing MetaCard Stacks

Moving Between Revolution and MetaCard

WhatÆs Next?

Summary

From MetaCard to Revolution

Revolution is based upon the proven MetaCard engine. The engine provides Revolution with the familiar object model, most of the language, and the code used to draw windows and controls on the various platforms supported by both MetaCard and Revolution.

The development environmentÆs user interface is somewhat different, with new menu items and windows. Revolution also adds some new capabilities via custom libraries. Fundamentally, however, Revolution works the way MetaCard does:

ò All object types supported by MetaCard are also present in Revolution.

ò The scripting language, Transcript, is identical in syntax and capabilities with MetaCardÆs language, MetaTalk.

ò Revolution can run onùand build standalone applications forùall platforms that are supported by MetaCard.

Revolution has an all-new development environment, especially designed for the needs of the cross-platform developer. New tools such as the Menu Builder, and the Geometry pane in the property inspector, automate repetitive tasks. The Database Query Builder lets you access Oracle, MySQL, PostgreSQL, and Valentina databases, as well as any ODBC-compliant database. And RevolutionÆs custom libraries offer additional functionality in areas such as database access and animation.

Importing MetaCard Stacks

The MetaCard and Revolution file formats are (and will remain) compatible. This means you can simply open a MetaCard stack in Revolution in order to start working with it in the Revolution development environment.

At times, the Revolution engine and the MetaCard engine may be out of sync. If you are using the very latest features of MetaCard, therefore, make sure you have the latest version of Revolution. The latest version is available for download at <<http://www.runrev.com>>.

To find out whether the Revolution and MetaCard engines you have are in sync, check the numbers returned by the version and buildNumber functions. If both functions report identical results in your copy of MetaCard and in your copy of Revolution, you can be sure that the engine versions of the two products are synchronized and that all the features of your MetaCard stacks—even those using the latest enhancements—will continue to work once you move the stack to Revolution.

Moving Between Revolution and MetaCard

If you wish, you can open a Revolution stack in MetaCard and edit it in the MetaCard environment. The basic features of the stack will work in MetaCard as in Revolution.

If you have trouble opening a Revolution file in MetaCard, use the following simple script in a button or the MetaCard message box:

```
answer file "Open which file?"  
if the result is empty then open stack it
```

A few issues exist when moving a Revolution stack back to MetaCard, and a few things may not work as expected. These are described in the following sections.

Custom libraries

If your scripts use commands or functions from Revolution custom libraries, those commands and functions will not work in MetaCard.

If you have used the Geometry or Property Profiles panes in the property inspector or the Animation Builder in developing your stack, the parts of your stack that depend on those features will not work in MetaCard. For example, if you have used the Geometry pane to specify how objects should be automatically moved when the user resizes the stack window, the objects will not move automatically when you resize the stack window in MetaCard. This is because these features use Revolution's custom libraries for their implementation, and those libraries aren't present in MetaCard.

(Menus developed with the Menu Builder will continue working in MetaCard, because the Menu Builder uses only features built into the engine, and doesn't use custom libraries.)

Tip: It's not necessary to use the command
start using stack "libURL"

when using Revolution. Revolution automatically includes its Internet library in the message path. When you create a standalone application, in Step 3 of the Distribution Builder window, make sure the "Internet Library" option on the Inclusions tab is checked. Your standalone automatically starts using the library when it starts up.

Appearance of text

Revolution and MetaCard use different default font settings, so if your stack's font and color properties are not set, the text and colors of your stack will appear somewhat different when opened in MetaCard.

Creating new objects

Some default object properties are different in Revolution and MetaCard, so if your scripts create new objects, their appearance and behavior may be different from what you expect.

Script style information

If you edit a script in MetaCard, any fonts, sizes, styles, and colors you've added to that script within Revolution will disappear. (The text and functionality of the script itself is unaffected by moving back and forth between the two development environments.)

What's Next?

Learning More About Revolution

As an experienced MetaCard developer, your next step is to familiarize yourself with the new user interface and the custom libraries.

Recommended tutorials

The first tutorial, "Getting Started", takes you through the process of creating and building a basic application. Although you will already be familiar with the basic concepts discussed in this tutorial, it will be helpful to invest an hour or less to go through the steps and create the application. Doing so will familiarize you with the different menu items and tools used to create applications, and make it easier to locate them in the future.

Going through the rest of the Tutorials will give you an overview of the Menu Builder, Animation Builder, Property Profiles, and Geometry tools. You'll find that these tools make standard application-building tasks faster and easier, automating much of the work.

The final tutorial, "Independent Study", introduces the Application Browser, which takes the place of MetaCard's Control Browser. (Much of the material in this tutorial deals with inheritance and application design and may already be familiar to you, so feel free to skip ahead.) You explore a sample application using the Application Browser feature and the Revolution script editor. After doing the Independent Study tutorial, you'll feel comfortable with the extensions and changes to the development environment, and fully ready to use your MetaCard experience to create Revolution applications.

Exploring the libraries

Revolution provides the following custom libraries:

Animation: Sprite animations; Animation Builder support

Database: SQL database access

Internet: HTTP and FTP protocols, uploading, and downloading

Printing: Printing fields and text directly

Profile: Managing sets of property settings for objects

Speech: Text to speech

Video: Capturing video from an input source such as a camera

XML: Parsing, manipulating, and building XML documents

Common: Various utility functions and commands

Revolution's Internet library is the same as MetaCard's "libURL".

The other custom libraries are unique to Revolution, and each one includes commands and functions that provide additional capabilities to your applications. (Some libraries are implemented in scripts, and some as externals.) Explore the custom libraries to find out what additions to the language Transcript provides.

Tip: To see a list of all the commands and functions in a particular custom library, open the Documentation window, click Transcript Dictionary, and choose the custom library you want from the menu at the top of the window.

Summary

In this topic, you have learned that:

- ò Revolution is based on the MetaCard engine, language, and object model, with a different user interface and some additional capabilities.

- ò You can use your MetaCard stacks in Revolution without any changes.

- ò If you move stacks back and forth between MetaCard and Revolution, make sure the same engine version is present in both products. To make sure the engine is the same, compare the version and buildNumber properties.

- ò Revolution custom libraries offer your applications added functionality in specialized areas like database access and XML.

About Revolution for SuperCard developers

See Also:

Getting Started Tutorial, altID property

This topic is for new Revolution developers who have experience with SuperCard and the SuperTalk language, and who want to learn how to leverage their SuperCard knowledge and existing projects to get a running start in Revolution.

A converter program is available to automate the process of converting SuperCard projects into Revolution stacks. You can start out by converting or re-creating your existing projects, then extend your work to include the new capabilities available in Revolution.

To fully understand this topic, you should know how to create stacks and write scripts using SuperCard. If you are an intermediate-level SuperCard developer, you have enough information to fully understand this topic.

Contents:

From SuperCard to Revolution

Converting SuperCard Projects

Common Traps for SuperCard Developers

Tips for Good Revolution Habits

WhatÆs Next?

Summary

From SuperCard to Revolution

As an experienced SuperCard developer, youÆll find the transition to using Revolution straightforward and easy. Just as you did in SuperCard, youÆll create objects in a stack window (which can have one or more cards). You write handlers in the Transcript language, whose syntax is very similar to SuperTalk.

Revolution supports all the object types SuperCard does, plus more: buttons, fields, images (bitmaps you can either paint in or display a file in), players, graphics, and scrollbars. QuickTime and QuickTime VR movies are displayed in the built-in player object, which is a full peer of other object types. You use the property inspector to change the appearance and behavior of these objects.

There are a few conceptual differences to be aware of:

- ò Each window is a stack object, and files can contain multiple stacks.
- ò You use groups instead of backgrounds to share objects between cards.
- ò You create a menu bar by creating a special group of menu buttons.

Stacks, projects, and windows

In SuperCard, each window is an object, and windows are contained in a project file. In Revolution, each window is still an object (called a stack), but there is no project object. Instead, each stack file contains one or more stacks, of which one is the main stack and any others are substacks. Like a project, the main stack is in the message path of all objects in the substacks. Unlike a project, the main stack is a stack in itself and has a window of its own.

The main stack opens by default when you open a stack file. Revolution applications usually use the main stack as either a splash screen or the main control window of the application.

Tip: Use the Geometry pane in the property inspector to have Revolution automatically move and resize objects in a stack when the user resizes the window.

Backgrounds and groups

In SuperCard, you can place objects in the background layer, and each card with that background displays those objects behind the card objects.

In Revolution, you use the Group menu item (or the group command) to combine any set of selected objects into a group. Unlike SuperCard backgrounds, groups can be added to any card in the stack, and a card can contain more than one group. (And unlike SuperCard graphics groups, Revolution groups can contain any object types, and grouped objects retain their scripts.) Groups can be interleaved with, or in front of, non-grouped objects.

Groups are very flexible. You can select and move a group, show it, hide it, give it borders or scrollbars, and place it on any card or remove it from any card. Like a background, a group can have a script. You can even create a group that contains other groups.

When you create a new card, any groups on the current card are placed on the new card, just like the way SuperCard adds a new card to the current background. However, it's important to remember that groups are not backgrounds. In SuperCard, every card belongs to a background. In Revolution, however, cards can exist without having any groups, and the groups belong to the cards they're on rather than vice versa.

Menus

In SuperCard, menu items and menus are special object types. In Revolution, all menus (including menus in the menu bar) are implemented as buttons.

To create a button menu, you set the style property of the button to menu, and set its menuMode property to specify a pulldown menu, popup menu (called an "option menu" in Revolution), or combo box. These button menus can be placed anywhere in a stack window.

To create a menu bar, you first create a button for each menu, setting the button's menuMode to "pulldown". Then you combine those menus into a group, placed at the top of the stack window. (This is the normal location for menu bars on Unix and Windows systems.)

If you set the stack's menubar property to the name of the group, Revolution displays the group's menus as a normal Mac OS menu bar and changes the window size to hide the buttons. This means that the menu bar will appear in the proper place for each platform: on Mac OS and OS X systems, the menus appear at the top of the screen in the menu bar, while on Unix and Windows systems they appear at the top of the stack window, in accordance with user-interface standards for those platforms.

Tip: Use the Menu Builder feature to automatically create and place the buttons for a menu bar, set its appearance appropriately for any platform, and specify keyboard equivalents for its menu items.

The development environment

Revolution's development environment is similar to SuperCard's. There is a message box (although the Revolution message box can execute multiple-line structures as well as one-line statements) and a tool palette for working with objects.

In SuperCard's Project Editor, you must switch between Design Mode and Run Mode before testing your scripts. Revolution's development environment is modeless: you can test a handler as soon as it's written, and use your stack within the development environment. To test your stack in a clean environment without any of Revolution's own menus, palettes, or other parts of the development environment, use the "Suspend Revolution UI" item in the Development menu. To prevent normal messages from being sent to your stack during editing, use the "Suppress Messages" item in the Development menu.

The development environment's menus, palettes, dialog boxes, and all is built entirely in Revolution. This not only demonstrates the power of the Revolution engine, it means that as you advance, there is the possibility of exploring the Revolution user interface and customizing it for your needs.

Converting SuperCard Projects

You can download a converter application for SuperCard projects from <http://www.runrev.com/revolution/sctorev/sctorev.sit>. The converter consists of a SuperCard project and a Revolution stack. Documentation is included with the converter program.

The process works like this:

1. Open the "SuperCard -> SIF Exporter" in SuperCard and export your project. The Exporter creates a set of text and image files.

Note: The project you're converting must have been saved in SuperCard 3.0 or later.

2. Open the "SIF to Rev importer.rev" stack in Revolution and import your exported files. The Importer creates a Revolution stack file from the text and image files.

Each window of your SuperCard project becomes a stack in the Revolution stack file that is created. Backgrounds in your project are converted to Revolution groups. Names and most other object properties are retained during the conversion.

The project script is placed in the script of a substack called "Directory of project name". To place this script into the message path, paste it into the script of the main stack.

All the menus from the project are placed in a separate stack. If your application will be made available for Unix or Windows, this stack will become a separate window containing the menu bar on those systems. You will need to decide whether your application design makes more sense with this detached menu bar, or whether the menus should appear inside your stack windows.

For most projects, the conversion process is straightforward. Following these guidelines will help ensure a smooth conversion:

Convert on a Mac:

The "SIF to Rev importer.rev" stack can be used to complete the second stage of the conversion on any platform Revolution supports. However, it is recommended that you run it on a Mac OS system,

particularly if your project contains special characters such as accented letters. The resulting stack file can then be used on any platform.

Use a project file:

The converter will not convert a standalone application created in SuperCard; you must use the original project file.

Remove quotes from object names:

Make sure the double quote character (") does not appear in any object names in your project before you convert it.

Un-nest nested groups:

The converter does not reliably handle nested groups of graphics, so if your project has groups that contain other groups, re-group them so they are one level deep before converting. You can nest groups in Revolution, so after the conversion is complete, you can restore the nesting if you wish.

Replace incompatible terminology in scripts:

A few common SuperTalk terms have different equivalents in Transcript. (See "Terminology Changes", below.) The converter does not alter your scripts, so you will need to make these changes by hand after converting.

Common Traps for SuperCard Developers

Revolution has a high level of compatibility with SuperCard, and almost all your knowledge will translate easily. However, there are a few minor differences in syntax, and many areas where Revolution offers a different method of doing things. Here are the most common gotchas and some useful tips:

Special characters in scripts

A few special characters (that require the Option key) can cause problems in scripts, if you move the stack to a Windows or Unix system:

• The SuperCard line-continuation character $\frac{1}{4}$ (option-L) will work in Revolution under Mac OS and OS X. However, it is not recognized under Windows or Unix, and a handler containing it will cause a script error. For cross-platform compatibility, use a backslash (\) instead. The backslash is recognized on all platforms.

• Similarly, the not-equal character \neq , the less-than-or-equal character \leq , and the greater-than-or-equal character \geq can be used on Mac OS and OS X systems, but not on Windows or Unix. For cross-platform compatibility, use the synonyms \ltgt , \lt , and \gt .

SuperCard character:	Use instead:
$\frac{1}{4}$ (option-L, option-Return)	
\neq (option-=)	\ltgt
\leq (option- \lt)	\lt
\geq (option- \gt)	\gt

Saving stacks

SuperCard saves changes to stacks automatically, but Revolution uses the more usual method of requiring the user to save changes explicitly. This has consequences in both the development environment and your applications.

In the development environment, your changes are not saved for you automatically. If you want changes saved automatically, one method is to write a plugin. Another is to simply include a `closeCard` handler with a save command in each stack you want to save automatically.

Revolution standalones cannot save themselves; they can only save separate files. (On Windows and Unix systems, a running application cannot modify itself. For consistency, Revolution also enforces this limitation on Mac OS systems.) This means that if the user enters data into your standalone and you want to save it, you'll need to provide a method, since data entered into any of the stacks in the standalone file itself will not be saved. There are two methods, and which one is best depends on what kind of data you want to save:

- ò If the user needs to save a small amount of data, such as settings or preferences, it is often simplest to save this data in a text file in a quit handler. Your standalone's startup handler can then read the data from the file and use it to set up the standalone's window by placing text in fields, setting checkboxes, or whatever else is needed.

Tip: If you want to store your application's settings in the Preferences folder, use the `specialFolderPath` function to locate the Preferences folder.

- ò If the user needs to save a larger amount of data—for example, in a database stack where the user is invited to make extensive notes—creating a stack file separate from your standalone is often the best method. This stack file serves as a document, owned by your standalone. Using this method, there is no need to move data into fields: when the separate stack file is opened, the data is already right there, ready to be modified and saved.

If users of your application need to save data, you will also need to provide this ability, since neither the Revolution development environment nor standalones created with it save stacks automatically. If you're creating your own custom menu bar, include a standard "Save" menu item in the File menu, or else make some other provision such as automatically saving windows when they're closed.

Custom properties instead of user properties

Revolution custom properties can hold any data you want to place in them, are saved with the stack, can be associated with any object, and are used in handlers just like built-in properties.

Unlike SuperCard's user properties, custom properties do not have to be defined before use. Simply setting the custom property creates it. Revolution also allows each object to have several sets of custom properties, which you can swap in and out as needed.

Tip: To see and edit an object's custom properties, select the object, choose Object menu Object Inspector, then choose "Custom Properties" from the menu at the top of the property inspector.

Overriding built-in commands and functions

In SuperCard, commands and functions pass through the message path, so a handler with the same name as a built-in command or function overrides its normal behavior.

In Revolution, built-in commands and functions are executed immediately rather than passing through the message path, so a handler with the same name as a built-in command or function will never be executed.

Everything in memory

When you open a stack file in Revolution, all its stacks are loaded into memory and kept resident until you purge the file. (Use the `destroyStack` property to remove files automatically from memory when you close them.) This makes for much faster access, since stacks are pre-loaded into memory and Revolution doesn't need to access the disk when you go to another card.

This also means that you cannot create stack files too large to fit into memory. If you have a large collection of data in your stack, you may want to split it into multiple stack files. Or keep media, such as pictures, videos, and sounds, in external files and display these items in referenced controls. (Files in referenced controls are loaded into memory only when you go to the card the control is on.)

Dates, times, and the start of the eon

The Mac OS date routines (and therefore SuperCard) use midnight, January 1st, 1904 in the local time zone as the "beginning of time". The `seconds` function counts from this moment. However, since Revolution is cross-platform, it doesn't use the Mac OS-specific date. Instead, time begins at midnight, January 1st, 1970 GMT. This will affect your stacks if you store or manipulate the value of the `seconds` function.

The `ticks` function in Transcript goes back to the start of the eon, not to the last time the system started up.

The date and time functions and the `convert` command use the U.S. format for dates and times by default. (You can use the system keyword to specify that you want the date and time formatted according to the user's preferences.)

Pathnames

In SuperCard, file pathnames use the Mac OS convention of separating levels with a colon, and a file path might look like this:

Hard Disk:Folder:File

Revolution uses the Unix style for pathnames, which uses slashes instead of colons, on all platforms to simplify writing cross-platform handlers. The same file path, written Revolution-style, looks like this:

/Hard Disk/Folder/File

The folder that contains the current folder is indicated in a relative path by `../`. For example, if the current folder is `Folder 1`, the path to `Folder 2` on the same level is written `../Folder 2/`.

Note: Revolution translates slashes to colons in pathnames, so if you have a file called `And/Or`, in a Revolution pathname it becomes `And:Or`.

Minor gotchas

Although almost all SuperTalk commands work in Transcript, there are minor syntax differences in a few of them. These may cause problems when using old SuperTalk code.

Visual effect quoting:

TranscriptÆs visual effect command does not allow the effect name to be enclosed in quotes. The statement visual effect "barn door" does not work in Revolution, though it works in SuperCard.

Tip: Revolution can also use QuickTime-based visual effects, which opens up many new possibilities. For more information, see the visual effect command.

Closing fields:

In Revolution, pressing the Enter key doesnÆt automatically close the current field. If you want to emulate this behavior, place an enterInField handler in the fieldÆs script (or the script of an object in the fieldÆs message path):

```
on enterInField
    select empty -- closes the field
end enterInField
```

Referring to card and background objects:

As noted above, Revolution uses groups instead of backgrounds (although in some situations you can think of groups and backgrounds as being almost the same). Revolution does not assume either the card or background layer when referencing buttons and fields. In other words, you can specify ôbutton myButtonö or ôfield ID 22ö and Revolution will find the button or field either on the card or in a group, without needing to be told which.

The active stack window:

In most applications, the active window holds the current document, and menu commands operate on the active window. In Revolution, because you can open stacks as palettes and dialog boxes, this is not necessarily the case and occasionally the active window is not the frontmost stack. Use the topStack function and defaultStack property to control which stack is the active window.

Graphics and images:

In Revolution, graphics and images are two distinct object types. A graphic is a resizeable shape, while an image holds a bitmapped picture. You cannot put bitmapped data into a graphic (although you can set a graphic to display a bitmapped pattern).

Graphic text:

Revolution does not have a graphic text tool. Instead, use a field to hold the text. You can set the fieldÆs showBorders property to false to remove its outline.

Scrolling windows and groups:

Revolution does not have a scrolling window type. However, groups as well as fields can have vertical and horizontal scroll bars. The simplest way to make a window scroll is to include all the windowÆs objects in a group, size the group to the size of the window (and set its lockLocation property to true to prevent it from automatically resizing), and set its vScrollbar property to true. You can also use this versatile capability to scroll a portion, or pane, of a window instead of the whole thing.

Terminology changes

Most SuperTalk terms can be used unchanged in Transcript. The most commonly-used exceptions are:

backSize property: Use the rectangle, width, and height properties of the group instead.

clickList message: Use mouseDown and mouseUp instead.

closeProject message: Use closeStack instead.

isNumber function: Use the is a operator instead.

itemSelect message: Use the menuPick message instead.

keyStroke and keyInField messages: Use the keyDown message instead. (To trap all keys including special keys use the rawKeyDown message.)

loc of the cursor: Use the screenMouseLoc property instead.

openProject message: Use openStack instead.

penBack property: Use the borderColor property instead.

setWindow command: Use the defaultStack property instead.

textHeightSum property: Use the formattedHeight property instead.

Tips for Good Revolution Habits

While your stacks will work in Revolution without implementing any of the suggestions below, they will be more efficient and work better cross-platform if you keep these tips in mind as you start to work in Revolution.

Replace externals with built-in commands and functions

Many of the externals commonly used by SuperCard developers can be duplicated in Transcript code instead. XCMDs and XFCNs can be used only on Mac OS systems, and need to be recompiled (and in some cases rewritten) if you want to bring your work to OS X, Unix, or Windows. Even though you can use HyperCard 1.x-format externals in Revolution, replacing externals with a script whenever possible will simplify your development task and ensure that your stack is compatible with other platforms without any problems.

Here are a few examples of tasks that SuperCard developers commonly use externals for, and the equivalent Transcript terms:

Task:	Transcript term:
Copy files and folders	revCopyFolder command revCopyFile command
Move files and folders	rename command
Delete files and folders	delete file command delete folder commands
Work with aliases	create alias command aliasReference property
Get a folder's contents	files function folders function
Read HTML pages	URL keyword htmlText property

Upload via FTP	ftp keyword libURLftpUpload command
Download via FTP	ftp keyword libURLftpDownloadToFile command
Use binary files	open file command (binary option)
Use binary data via serial	open file command (binary option) modem: keyword printer: keyword open driver command
Work with resources	getResource function getResources function setResource function copyResource function deleteResource function
Display a custom dialog	modal command
Display a palette	palette command

Replace idle handlers with delayed sends

At times you will need to execute one or more commands periodically. In SuperCard, it is common practice to place such commands in an idle, mouseStillDown, or mouseWithin handler.

While Revolution supports idle, mouseStillDown, and mouseWithin, it's more efficient to use the send command to send a message at a specified future time. This means you can place the commands to be executed in a handler, and at the end of that handler, send a message to execute it again after the delay you specify.

For example, suppose you want to annoy the user by beeping every ten seconds. In SuperTalk, you might use this idle handler, which uses a global variable to keep track of the next beep time:

```
on idle -- SuperTalk example
  global nextBeepTime
  if the seconds _ nextBeepTime then
    beep
    put the seconds + 10 into nextBeepTime
  end if
end idle
```

In Revolution, the send command has an additional option to send a message after a delay you specify. With this capability, you can execute a handler, and within that handler, send a message to execute the same handler again after ten seconds. This example annoys the user similarly to the idle handler above, using the send command instead:

```
on annoyUser -- Transcript-friendly equivalent
  beep
  send "annoyUser" to me in 1 second
end annoyUser
```

This approach offers greater efficiency, since Revolution does not need to continually execute an if...then statement in an idle handler to determine whether enough time has passed. It also gives you

much greater control over timing. Use of the idle handler to perform lengthy tasks may cause Unix systems to lock up, and this alternative technique avoids such potential problems.

The one major difference between the two examples is that the process does not start automatically. To start annoying the user with beeps, you send the `annoyUser` message to the object that contains the handler, either in another handler or in the message box. Once the first `annoyUser` message is sent, the `send` statement inside the handler automatically queues the next one.

Tip: To cancel a pending message that has been sent with a time delay, use the `cancel` command.

Use `mouseMove` instead of polling the mouse state

When you need to monitor the mouse's position, it is usual in SuperCard to poll the mouse position within a repeat loop (or in an `idle` or `mouseWithin` handler):

```
on mouseUp -- SuperTalk example
  repeat while the mouseLoc is within the rect of me
    put the mouseLoc into field "Mouse Position"
  end repeat
end mouseUp
```

As with the `idle` and `mouseWithin` messages, this kind of construction is supported by Revolution. However, it requires Revolution to check the mouse position continually, even when it hasn't changed. It's more efficient to use a `mouseMove` handler in an object's script to track the mouse's motions while the pointer is within the object.

```
on mouseMove -- Transcript-friendly equivalent
  put the mouseLoc into field "Mouse Position"
end mouseMove
```

Use URLs instead of open file

You can use URLs as containers, and use commands like `get` and `put` with them. This applies to URLs of all types: `ftp` and `http` URLs (which are used to upload and download files), as well as `file`, `binfile`, and `resfile` URLs, which can be used to put data into or get data from local files.

For example, to read a file's contents into a field, you might use this set of SuperTalk statements:

```
open file "Disk:Folder:File" -- SuperTalk example
read from file "Disk:Folder:File"
put it into field "File Contents"
close file "Disk:Folder:File"
```

You can use a similar sequence of statements in Revolution (after changing the pathname to use slashes instead of colons), but you can also use this simple statement to replace all four lines:

```
put URL "file:/Disk/Folder/File" into field "File Contents"
```

What's Next?

Learning More About Revolution

As an experienced SuperCard developer, your next step is to familiarize yourself with the new user interface and the custom libraries.

Recommended tutorials

The first tutorial, "Getting Started", takes you through the process of creating and building a basic application. Although you will already be familiar with most of the concepts discussed in this tutorial, it will be helpful to invest an hour or less to go through the steps and create the application. Doing so will familiarize you with the different menu items and tools used to create stacks, and make it easier to locate them in the future.

You should also make time to go through the second tutorial, "The Menu Builder". The process of creating menus is similar in Revolution and SuperCard, but the underlying concepts are somewhat different, so a little time spent on this subject will help avoid confusion in the future. You may also want to read through the topic "About menus and the menu bar" to obtain the conceptual background underlying the Menu Builder.

The other tutorials will also be useful to you in learning about special Revolution features. In particular, the final tutorial, "Independent Study", includes a full Revolution application with heavily-commented code.

Learning from the Cookbook

The Transcript Cookbook is a section of the documentation that contains annotated code "recipes" for doing various tasks.

As an experienced SuperTalk programmer, you can look through the Cookbook to see Transcript code in action and get a better feel for the differences between good SuperTalk practice and good Transcript practice. You'll find that some of the recipes are written the same way you would write them in SuperTalk, while others illustrate different methods and new capabilities. For example, the topic "Recipe for a card slideshow" demonstrates the use of delayed sends for a task that you would probably use "idle" for in SuperTalk.

To see the Cookbook, click "Transcript Cookbook" in the main Documentation window. Or choose any topic starting with "Recipe for..." from the Development Guide section of the documentation, or from the See Also menu in any documentation topic.

Summary

In this topic, you have learned that:

- ò Revolution's object model and development environment are similar to SuperCard's. Its language, Transcript, is similar to SuperTalk, though larger with more commands, functions, and properties.

- ò You can have one or more stacks in a file. Each stack can be displayed as a window, modeless dialog box, modal dialog box, or palette.

- ò A group is a set of objects that have been combined into one. The group has its own script, and so does each of its objects. Groups can be placed on a single card or shared between multiple cards.

- ò All menus in Revolution are created from buttons; a menu bar is created from a group of buttons. The Menu Builder automates the task of creating a menu bar.

ò You can use the converter at <<http://www.runrev.com/revolution/sctorev/sctorev.sit>> to convert a SuperCard project to a Revolution stack.

ò You can use HyperCard 1.x XCMDs and XFCNs in Revolution. However, in most cases, externals are not needed because the functionality of the most common externals is already built into Revolution: file handling, URL and Internet access, and so on.

ò Most of your SuperTalk code will go on working in Revolution, but for certain constructs, Revolution offers a better or more efficient way of writing the same functionality. Look through the Transcript Cookbook to see code examples and explanations.

About Revolution for new developers

See Also:

Supported Platforms Reference, Getting Started Tutorial, Recipe for Hello World

This topic is for new Revolution developers who have little or no programming experience, and who need an overview of Revolution's capabilities and an introduction to basic programming concepts so that they can get started creating applications.

Beginning developers should also go through the Revolution Tutorials in order to get hands-on experience in developing a simple Revolution application. You may either read this topic first and then do the "Getting Started" tutorial, or do the tutorial first, or leave both windows on the screen and switch between them.

Contents:

Starting Out in Revolution

Stacks and Objects

Introducing Programming With Transcript

What's Next?

Summary

Starting Out in Revolution

Traditionally, creating software has been a difficult and painstaking task that could be performed only by experts using powerful but arcane programming languages. As computers became easier to use, they became more difficult to program. The basic elements of a modern user interface—windows, menus, buttons—had to be created by entering complicated instructions. A deep knowledge of operating system internals was required in order to write even the simplest applications.

Revolution changes all that. Revolution provides a set of basic building blocks—like push buttons, windows, and text fields—that you put together to create your application. It automatically handles basic behaviors like typing text and highlighting buttons, so you can concentrate on the essentials of your application. The Transcript language, while a fully capable programming language in its own right, is easy to read and easy to get started in.

Even if you're not a formally trained programmer, you can become productive in Revolution quickly. If you've never done any programming at all, you will find it's easy to start by laying out a user interface and creating simple scripts, then moving on to more complex functionality as your skills develop.

Stacks and Objects

The basic building blocks of a Revolution application are called objects. Revolution objects include stacks, cards, controls, and groups.

The building blocks: objects

The first step in creating a Revolution application is creating a window, which in Revolution is called a stack. Each window you see in Revolution is a stack. Palettes, dialog boxes, and standard windows are all stacks.

Each stack contains one or more screens of information, called cards. Each card can have a different appearance, or all the cards in a stack can look the same. By going from card to card in a stack, you change what's being displayed in that stack's window. You can think of a Revolution stack as a stack of playing cards (hence the name), where you can flip through the cards, but only one card at a time is visible. A stack can have a single card or many cards.

Once you've created a stack window, you can start drawing the buttons, text fields, and other controls that make up your application. Revolution operates very much like a draw program: you can select controls by clicking them, move them by dragging them around, resize them, and move them backward and forward.

You can also group controls together if you want them to operate as a set. For example, if you have a set of navigation buttons that go from card to card in your stack, you can make them into a single group. Groups can appear on more than one card, so your navigation buttons can appear on each card of your stack.

Changing objects: properties

A property is an attribute of a Revolution object. Each type of object has properties that determine the object's appearance and behavior. For example, a stack window's location on the screen is determined by the stack's location property. If you want to find out exactly where the window is, you can check the location property. And if you want to move the stack, you can change its location.

Every aspect of the object's look and feel is its color, size, position, and more is determined by the object's properties. Whether a button has an icon, whether a stack window is visible, and whether a text field has a scrollbar are all determined by the properties of those objects. An object may have over a hundred different properties, depending on what kind of object it is.

Changing properties:

You access an object's properties by selecting the object, then choosing Object menu Object Inspector. The property inspector contains a menu at the top that lets you choose to show various groups of properties, including the object's custom properties.

You can see and change the properties using this palette. (You can also change properties in a handler that's part of the Transcript code of your application. We'll talk about that below.)

Tip: The buttons and text boxes in the property inspector are labeled with user-friendly, descriptive names to help you understand the effect of changing each property. To see the Transcript property name, hold the mouse over the button or text box until a tooltip appears.

Ownership and inheritance

You can see, from the description of object types above, that each object is part of another kind of object. Controls such as buttons and fields are part of cards. Cards are part of stacks. If controls are grouped together, each control is part of the group, and in turn, the group is part of the card it's on.

When an object is part of another object, the second object is said to own the first object.

The object types can be arranged in an object hierarchy, with each object being owned by one on the level above it, another object owning that one, and so on. (For example, a button's object hierarchy

includes the button itself, the card the button is on, and the stack that the card is in.) This hierarchy is important in two ways:

- ò Some properties of an object, such as its color and text font, take on the settings of the object above it in the object hierarchy. For example, if a field's background color property is not specified, it takes on the background color of the card that owns it. If no background color is specified for the card either, the stack's background color is used.

This means you can set a background color for a stack, and every object in it will automatically use that background color without your having to set it for each object. This process of checking first the object, then the object's owner, then the object that owns that object, and so on, is called inheritance. Each object inherits the background color (and other color and font-related properties) of the object above it in the hierarchy.

- ò The program code you write for an object, similarly, can be inherited from the objects above it in the hierarchy. For example, if you want all the buttons on a card to do the same thing, you can write the code just once—for the card—and all the buttons on the card can use the code.

Introducing Programming With Transcript

Programming in Transcript is not trivial, but with some information about the basics, it's easy to get started writing real programs. As you gain practice and experience, you'll be able to construct more and more complex code to do more and more tasks. To give you a running start, here are some basic programming concepts and how you use them in Transcript.

Tip: Scan the Transcript Dictionary. Look at the examples at the top of each topic to see how each Transcript term is used in a program statement. If you see a technical term whose meaning you don't know, click it to see whether there's a Glossary entry that explains the term.

To see an object's script, select the object and choose Object menu Object Script. A script consists of modules called handlers, each of which responds to a message.

Messages

A message is an announcement that an event has occurred, made to the object the event affects. Events include user actions (such as typing a key or clicking the mouse button) and program actions (such as completing a file download or quitting the application). Revolution watches for events and sends a message to the appropriate object when an event occurs.

For example, if you click a button, Revolution sends a message called `mouseDown` to that button. If the button's script contains program code for the `mouseDown` message, that code is executed. The program code that goes with a particular message is called a handler for that message.

Handlers

A typical `mouseDown` handler might look like this:

```
on mouseDown
    beep
    go to next card
end mouseDown
```

The first and last lines of this message handler specify that this is the beginning and end of the code that will be executed when the button gets a mouseDown message. All message handlers begin with the word `on` and the message name, and end with the word `end` and the message name. The lines between are the actual code that is executed. Each line is a statement, which is an instruction to Revolution to do something. When you click this button, you're telling Revolution to beep and then visit the next card in the stack.

You can see that Transcript code looks very much like English. This makes it easier to understand and to read than most programming languages. However, it's important to remember that this is programming, not English. If you put the words of a sentence in a slightly wrong order, other people usually can understand what you mean, but computers are less flexible, so you need to be careful to make sure your code is correct and that there are no misspellings or other minor mistakes.

What if the button doesn't have a handler for the mouseDown message? This is where inheritance comes in. If the button does not handle the mouseDown message, Revolution passes it on to the object that owns the button. If that object doesn't have a mouseDown handler either, the mouseDown message continues to be passed up the object hierarchy. If none of the objects has a mouseDown handler, nothing happens.

This kind of handler, a message handler, reacts to messages. One script can have more than one message handler; each handler reacts to a different message. Scripts can also contain other kinds of handlers—function handlers for computing values, and `getProp` and `setProp` handlers for custom properties. See the [Commands and functions](#) and [Properties, custom properties, and property sets](#) topics for more information about these kinds of handlers.

Commands and functions

A command is an instruction to Revolution to do something. Every program statement starts with a command.

In the example above, we used two commands: `beep` and `go`. Transcript has over a hundred and fifty built-in commands, all of which are described in the Transcript Dictionary. Here are some examples of how commands are used in a statement:

```
beep
answer "Hello world!"
create folder "New Folder"
```

Tip: You can use the message box to try out any command without putting it into a handler. To open the message box, choose Tools menu Message Box. Then type in any of the example lines above, and press the Return key to see Revolution execute the command.

You probably noticed that the statement with the `go` command contained more than the command name: it also contained a description of where to go. These extra pieces of information in a statement are called the command's parameters. Parameters specify details of how the command operates. The name `New Folder` in the example with the `create folder` command is also a parameter, and specifies the name of the folder to create.

Commands, then, perform an action. Functions, on the other hand, tell you a result. For example, the `sum` function gives you the sum of a list of numbers:

```
sum(12,22,3)
```

Transcript has almost two hundred built functions covering everything from trigonometry to the computer's Internet address.

Unlike commands, functions don't actually do anything—they just provide a result—so you have to use them in a statement along with a command that does something with the result the function provides. For example, this statement uses the put command along with the time function:

```
put the time into field "Clock"
```

Variables

A variable is a container for a value. Using a handler, you can put anything you want into a variable—numbers, words, long chunks of text, even the contents of a file—and change what's in the variable at any time.

The variable does not appear anywhere on the screen, but you can use its contents in your handlers. Whenever you use the variable's name in a Transcript statement, Revolution substitutes whatever is in the variable.

For example, this command displays "Hello world!" in a dialog box:

```
answer "Hello world!"
```

Now suppose you want to use the same dialog box in various places in your handler. You can put the "Hello world!" text into a variable:

```
put "Hello world!" into whatToSay
```

This statement puts the text "Hello world!" into a variable called whatToSay.

Later in the handler, when you use the answer command with the variable, Revolution substitutes what's in the variable for the variable's name. Revolution turns this:

```
answer whatToSay
```

into this:

```
answer "Hello world!"
```

To change the words that appear in the dialog box, you only need to change it once, in the line starting with the put command.

Control structures

At times, you may want to execute statements only if a certain condition obtains. For example, you might want a handler to beep, but only if the mouse button is being pressed. The parts of a programming language that control which statements are executed, and in what order, are called control structures.

Transcript includes several kinds of control structures to use for different purposes. The two control structures you'll probably use most are if...then (to execute statements only if a condition is true) and repeat (to execute a set of statements more than once).

Here's an example that uses the if...then control structure:

```
wait 3 seconds
if the mouse is down then
    beep
end if
```

This example waits 3 seconds, then checks whether the mouse button is still being pressed. If it is, the computer beeps. If the mouse button is not down, though, Transcript skips everything between the if statement and the end if.

The repeat control structure executes a set of statements more than once. You can repeat for a specified number of times, or keep repeating over and over until a condition you specify becomes true. This example keeps going from card to card until you click again to stop it:

```
repeat until the mouseClick
    go to next card
end repeat
```

Tip: You can use the multiple-line mode of the message box to try out control structures, or other sets of statements that need more than one line. To use the multiple-line message box, click the second icon from the left at the top of the message box. Then type in any of the examples above, and press the Enter key to see Revolution execute the statements.

What's Next?

Going Further With Revolution

Next, try the Getting Started Tutorial, which will give you a hands-on grounding in the process of building a simple application and writing scripts for it.

The Independent Study Tutorial should be your next stop. It provides a sample application and tells you how to study its scripts. The scripts in the sample application contain plentiful comments to explain what each line does.

As you continue working with Transcript, get into the habit of looking up terms in the Transcript Dictionary. The dictionary includes every Transcript command, function, operator, property, and control structure, with information on how you use each of them in a statement. Scanning the Dictionary can be very helpful, even if you don't remember everything you read, because it will give you an idea of what is possible with Transcript and where to look for a feature.

Tip: When you see a boldfaced Transcript term in the documentation, clicking it opens the Transcript Dictionary to the entry for that term.

Once you have your feet wet in Revolution, the best way to learn more is to build an application for yourself. Decide on a simple application to meet some of your own needs—a searchable address book, a digital clock, or whatever appeals to you—and write it from scratch. Or re-implement a simple

application you already have and use—maybe one you’ve always wished worked a little differently or had an additional feature.

Summary

In this topic, you have learned that:

- ò Revolution provides a set of building blocks—buttons, text fields, windows, dialog boxes, and more—that you put together to create an application.
- ò The Revolution building blocks are called objects. Every Revolution window is a stack object. Stacks can contain multiple cards (views), and each card can contain other objects such as buttons, fields, and scrollbars.
- ò You change the appearance and behavior of objects by changing their properties. You can change any property in a script, and set most properties interactively using the object’s property inspector.
- ò Revolution’s programming language, Transcript, is English-like and easy to read and understand. You can start by writing simple routines, called handlers, and progress to writing more complex handlers as your knowledge grows.
- ò To learn more about Revolution, work through the tutorials, scan the Transcript Dictionary, and experiment with creating a simple application.

About Revolution for experienced programmers

See Also:

Supported Platforms Reference, Getting Started Tutorial, Independent Study Tutorial, Recipe for Hello World

This topic is for new Revolution developers who have experience creating applications in traditional programming languages (such as C, C++, or Java), and who require an overview of RevolutionÆs development process and programming model.

To fully understand this topic, you should understand the basics of procedural programming (subroutines, functions, conditionals, and variables). You should also understand the concepts of modern user-interface design (windows, dialog boxes, and menus). If you have written a few simple applications in a compiled procedural language for a modern operating system, you have enough information to fully understand this topic.

Contents:

Moving Into Revolution

The Revolution Development Environment

Objects in Revolution

The Transcript Language

WhatÆs Next?

Summary

Moving Into Revolution

Revolution is a development environment that integrates user-interface design with writing and testing program code. Instead of building your user interface in one environment and writing your code in another, then combining them, you create the user interface right in the development environment, using drag-and-drop tools to draw objects right onto your applicationÆs windows, then write code for your objects.

While Revolution and the Transcript language are full-featured and capable of complex operations, the learning curve is gentle. Experienced programmers will begin to be productive with Revolution in just a few hours after doing the tutorials.

Object-based development

Every user-interface element in a Revolution application is an object: windows (including palettes and dialog boxes), push buttons, text fields, media players, bitmap images and vector graphics, scrollbars, and more. You modify the appearance and behavior of these objects in the property inspector.

Then you write code in the Transcript language for each object, associating the code directly with the object. (For example, the code affecting a button is contained in the button, and goes with it when you copy and paste the button.)

You can change an objectÆs properties directly from within a Transcript program statement, as well as in the property inspector in the development environment. This means your application can change the appearance and behavior of the user interface at runtime.

The development environment provides all the basic user-interface elements—menus, windows, buttons, text fields, and so on—and pre-defines much of their standard behavior. For example, text editing, text drag and drop, button highlighting, and window zooming, minimizing, and maximizing are all automatic behaviors of their respective objects. (You can override the automatic behavior, in most cases, by changing properties and intercepting messages.)

Very-high-level language

Transcript, Revolution's scripting language, is of the type known as a very-high-level or fourth-generation language (4GL). This means that it operates at a higher level of abstraction than more traditional programming languages such as C or Java, and that it handles many details of programming for you.

Transcript includes these typical very-high-level-language aspects:

- ò English-like syntax: Transcript is easy to read and understand, and largely self-documenting (especially if you choose variable and procedure names well).
- ò Typelessness/dynamic variable typing: Except for the distinction between arrays and scalars, variables are typeless. Transcript automatically handles conversion between strings, integers, floats, and other data types as needed.
- ò Interactive interpretation: As soon as you change the code in any object, the changes are ready to be executed. Code is silently compiled when you close the script editor. You don't need to build and run the application to test code changes.
- ò Compression and abstraction: Transcript can express complex operations in a single statement. For example, sorting the lines of a variable, writing binary data to a file, and displaying a file dialog box filtered by file type can each be done with a single line of code.

Message-driven programming

The first computer programs were sequential. Their interaction with the user proceeded in a straight line: the program presented a prompt and waited for the user's answer, and the user could offer input only when asked by the program. The program controlled the action the user could take.

Modern user interfaces require a different style of programming. Users can click a button, choose a menu item, close a window, and take other actions in whatever order they choose, and the program reacts to the user's actions instead of controlling them. To accommodate this more flexible user interface, a new programming technique is required: any user action—clicking the mouse, typing a key—is an event, and the program is built to respond to events.

Event-driven programs have a central event loop, which iterates continuously and watches for events. An event is a user action (such as a mouse click) or a system action (such as the arrival of data on a serial port). When an event occurs, the next iteration of the program's central loop notices it and responds accordingly.

Revolution is inherently event-driven. However, there is no event loop or `ômain()ö` function. All Transcript code is contained in message handlers, which are executed when the object whose script contains the handler receives the corresponding message.

The Revolution engine watches for events and automatically sends the correct messages. Whenever the user does something, Revolution sends the corresponding message to the object that the action was performed on. For example, when the user clicks a button, Revolution sends a `mouseDown` message to the button.

Program initialization is also done in message handlers: Revolution sends messages when the application starts up and when windows open, and you can write handlers for these messages to initialize the application environment as needed.

Messages are sent either by Revolution, in response to user actions or system events, or by other handlers. You can also send your own custom messages to any object, and write custom message handlers to respond to these messages.

Integrated runtime environment

There is no compile/link/run cycle; the test runtime environment, the user interface builder, and the program editor are one and the same. You go directly from editing code to modifying the user interface to testing the application and back again, without any need to re-launch or switch environments. (To test your application in a clean environment without any of Revolution's own menus, palettes, or other parts of the development environment, choose Development menu Suspend Development Tools.)

When you've completed testing, you compile your application into a double-clickable standalone using the Distribution Builder. You can build standalones, from the same file, for any or all of the platforms Revolution supports. (Revolution Express supports building applications only for a single platform.) Your finished applications automatically display the native appearance and behavior for each platform you deploy on.

The Revolution Development Environment

As an integrated development environment, Revolution contains all the tools you need to create applications: interface builder, code editor and debugger, runtime environment, and application builder. The development environment's menus, palettes, dialog boxes, and all is built entirely in Revolution. This not only demonstrates the power and speed of the Revolution engine, it means that as you advance, you can explore the Revolution user interface and even customize it for your needs.

These are the basic items you can expect to use most often when creating Revolution applications. All these tools and their uses are described in the Revolution Tutorials, but here is a quick overview:

The Browse and Pointer tools

Like a draw program, Revolution's interface builder has various modes you access by clicking tools in a tool palette. Choosing Tools menu Tools Palette shows the tool palette. The two tools at the top are the ones you will use most often.

The tool on the left, called the Browse tool, is shaped like a hand. You use the Browse tool to test your application, to click buttons, and to enter text into fields.

The tool on the right, called the Pointer tool, is shaped like an arrow. You use the Pointer tool to select objects in your application's windows. To select an object such as a field or button, you click the object with the Pointer tool.

The property inspector

A property is an attribute of a Revolution object. Each type of object has built-in properties that determine the object's appearance and behavior. (For example, an object's size is determined by its height and width properties.)

You can also define custom properties for any object, and use them to store any kind of data you choose. Like built-in properties, custom properties are stored with the object.

You access an object's properties by selecting the object, then choosing Object menu Object Inspector. The property inspector contains a menu at the top that shows different panes for various groups of properties, including the object's custom properties.

You can also set any property programmatically using Transcript's set command.

Tip: To see a list of all the properties for a particular object type, open the Documentation window, click Transcript Dictionary, and choose the object type from the menu at the top of the window.

The script editor

You use the script editor by selecting the object and choosing Object menu Object Script. Each object has its own script, which consists of the Transcript code that belongs to that object. A script may contain one or more individual routines, called handlers.

The Revolution script editor accepts styled text, so you can use colors and fonts to emphasize portions of your code, mark unfinished code, and so on. To search a script, choose Script menu Find and Replace while in the script editor.

An object's script is one of its properties, so you can change an object's script to change the code that belongs to it programmatically, the same way you set any other property.

The message box

The message box is a small palette that serves as an instant command interpreter, a mini-console, a calculator, and a quick command tester. You can enter any command into the message box for immediate execution, or enter any expression for evaluation.

Tip: Try entering an arithmetic expression such as $2 + 2$ into the message box, and press Return to evaluate it.

To open the message box, choose Tools menu Message Box.

The Application Browser

The Application Browser gives you a hierarchical view of all the objects in every open Revolution file. Here you can browse all open files, access each object's properties and scripts, and delete objects.

To open the Application Browser, choose Tools menu Application Browser.

The Distribution Builder

You use the Distribution Builder to create your finished standalone application. Using the Distribution Builder, you can create applications for any supported platform from a single file or a set of files. (Revolution Express supports building applications only for a single platform.)

Note: Because of the special file system requirements of Mac OS applications, you can build for Mac OS only on a Mac OS or OS X system.

To open the Distribution Builder, choose File menu Build Distribution.

The development process

The following is a quick overview of the basic process of developing an application in Revolution. For more concrete, hands-on information, do the Revolution Tutorials.

1. The first step in developing a Revolution application is creating a new main stack. Typically, this stack will be your application's main window.
2. Next, create any other windows, palettes, and dialog boxes your application needs, as substacks of the main stack. This allows your entire application to reside in a single file.
3. Next, create the controls your application needs in each window. You can create push buttons, popup menus, text fields, list fields, and more. Use the "New Control" submenu in the Object menu to get an idea of the kinds of control you can create.
4. Use the Menu Builder (in the Tools menu) to create a cross-platform menu bar for your application.
5. Finally, write the code needed for each object to implement its behavior, including library routines as needed.
6. When you're done, choose File menu Build Distribution to create standalone applications for the platforms you want to deploy on. The Distribution Builder creates a native application for each platform you select.

This development process is "inside out" compared to the conventional development methodology. In Revolution, it's usually most efficient to first develop the user interface, then write the code for each interface element. (Of course, you can refine, change, and add to both user interface and code at any stage in the development process.)

Your Revolution application will generally be cross-platform-ready as designed, though a few tweaks may be needed for optimal function and display on all platforms. Transcript language elements that vary between platforms are marked in the Transcript Dictionary. You can preview the look and feel of your application on any platform by choosing View menu Look and Feel.

Objects in Revolution

Revolution programming is object-based. There are twelve classes of objects, arranged in a well-defined object hierarchy which uses the following rules of behavior:

- Objects inherit the behavior of other objects higher in the hierarchy.
- Program execution is triggered by messages, which are either dispatched by the Revolution engine in response to user actions and other events, or sent by one object to another object.

ò All but two of the object types (video clips and audio clips) have a visual representation. You can hide any object, but all objects are user-interface elements, at least potentially.

For details about each object type, see the topic ôAbout object types and object referencesö.

Object relationships

Objects are related to each other in an ôis-part-ofö model, rather than an ôis-aö model. Each object is part of another object type thatÆs at a higher level of the object hierarchy. For example, buttons inherit from the windows they belong to, but a button is part of a window, rather than being a type of window.

Messages pass from object to object according to the rules of the object hierarchy: if an object does not handle a particular message, the message is passed to the next level of the hierarchy.

The object types

A window in Revolution is called a stack. Each window you see in Revolution is a stack. Palettes, dialog boxes, and standard windows are all stacks. Stacks occupy the highest level in the object hierarchy: every other type of object is contained in a stack.

Each Revolution file contains one or more stacks. The first stack in the file is called the main stack, and is part of the object hierarchy for all other stacks in that file. Any other stacks in the same file are substacks of the main stack. If a stack file contains only one stack, the stack is a main stack.

Each stack, whether it is a main stack or a substack, contains one or more screens of information, called cards. The card is the entire content area inside the stack window, and a stack window displays one card at a time. Each card can have a different set of objects, or all the cards in a stack can contain the same objects.

If a stack contains more than one card, the cards have a specific order: there is a first card, a second card, and so on. You can use Transcript commands to navigate between cards or to go directly to a specific card.

Each card, in turn, can contain a variety of objects: buttons, text fields, bitmapped images, vector graphics, scrollbars, video or sound players, and (on Unix systems) EPS objects. These objects are referred to as controls because the user interacts with them.

Any set of one or more controls can be incorporated into a group. When a control is included in a group, the group becomes a part of the object hierarchy for that control. Groups can also be nested, so a group can contain one or more groups. A group can appear on more than one card in a stack.

Finally, audio clip and video clip objects hold sound or movie data. Neither audio clips nor video clips appear in the stack window. Both audio clip and video clip objects inherit from the stack they are in, and are not part of a card.

Passing messages between objects

If a message is sent to an object, that object can either handle the message or ignore it. If a message is ignored, it proceeds to the next object, in search of a message handler for the message. The message path is the set of rules that determine which objects, in which order, have the opportunity to respond to a message.

Each message has an initial target, which is the object the message is originally sent to. For example, if the user clicks a button, a `mouseDown` message is sent to the button that was clicked. If that button does not handle the message—that is, if the button doesn't have a `mouseDown` handler in its script—the message passes to the current card, which is the next object in the object hierarchy. If the card does not handle the message either, it passes to the next object in the hierarchy, and so forth, until the top of the hierarchy is reached.

The `pass` and `send` commands can be used to temporarily override the normal message path. You can also use the `insert script` and `start using` commands to insert objects into a specified point in the message path. For example, you can make an object into a library by putting appropriate handlers in its script, and use the `insert script` command to make those handlers available to every object in your application.

Revolution object handling compared to object-oriented languages

If you have some experience with an object-oriented language such as C++, much of Revolution's handling of objects will be familiar to you, with some differences in terminology. In Revolution, you instantiate (create) an object—for example, a button. You give it data (set its properties) and methods (create handlers in its script).

There are also some differences:

- ò Each Revolution object has physical existence in a stack file. Objects, once created, are persistent and don't need to be explicitly instantiated every time you initialize the application (more precisely, the Revolution engine instantiates the stored objects when reading a stack file), although you can create new objects at run time if needed.

- ò All objects are part of the user interface and, except for audio clips and video clips, all have a visual representation on the screen. It's easy to select, examine, and debug an object.

- ò Classes (object types) and their relationships are pre-defined: you create specific objects, but you cannot create a new kind of object.

Tip: If you need a custom object, the easiest way is to start with an existing object type, adjust its properties to suit your needs, then use this customized object as a template to create more copies as needed. To create a compound custom object, the best way is to create a group that includes the parts of the custom object.

- ò Revolution does not restrict data exchange between objects. Any object can get and set properties of any other object, as well as send messages to any other object.

The Transcript Language

Transcript is a verbose language with an English-like syntax, allowing for easy readability and maintainability. Here's an example of a routine that responds to a keystroke by beeping if the pressed key is anything other than a digit:

```
on keyDown theKey
  if theKey is a number then pass keyDown
  else beep
end keyDown
```


Quick facts about Transcript

The Transcript language is case-insensitive.

The language size:

Coming from a traditional language such as C, C++, or Java, you'll find Transcript to be large in terms of the number of reserved words it contains, with well over a thousand commands, functions, properties, and built-in messages. There are two reasons for this:

- ò Transcript controls a user interface, so there are many object properties, and some commands, that deal directly with that user interface.

- ò Many commands are included in Transcript that, in another language, might be implemented as library functions instead. (For example, sockets are part of the core language rather than a separate library.)

It may be useful to think of Transcript as a core language, plus extensive GUI library, plus utility libraries, although it is not actually implemented this way.

Compilation:

Scripts are internally compiled to a byte-code-like representation, which combines the speeds characteristic of a compiled language with the flexibility of an interpreted language.

Compilation is done automatically when you close the script editor.

Statement delimiters:

Statements must be separated by either returns or semicolons; braces and parentheses are not used to structure code.

Transcript scripts are not otherwise whitespace-sensitive, although the script editor can auto-indent scripts to display their structure.

Tip: To auto-indent the current handler in the script editor, press the Tab key.

Script structures

Each object has a script consisting of one or more routines, called handlers. There are four handler types:

- ò Message handlers are executed when a particular message is received by the object. (Messages can be sent either by Revolution, in response to user actions or system events, or by other handlers.) They normally do not return a value to the caller, and are the Transcript equivalent of procedures or subroutines. Message handlers begin with the word `on`.

- ò Function handlers are executed when a function call is made by a handler in the same object or lower in the object hierarchy, and typically return a value to the calling handler. Function handlers begin with the word `function`.

- ò SetProp handlers are executed when a particular custom property is set for the object or for an object lower in the object hierarchy. SetProp handlers begin with the word `setProp`.

ò GetProp handlers are executed when a particular custom propertyÆs value is requested. GetProp handlers begin with the word getProp.

The script of a particular object can contain any or all of these handler types, and can contain an effectively unlimited number of handlers of each type.

Calling subroutines and functions

Transcript distinguishes between subroutines, which are implemented as message handlers, and functions, which are implemented as function handlers.

You call a message handler by using its name (and any parameters) as a Transcript statement. The message handler can reside in the script of the same object as the calling handler, or in any other object. The message is sent through the normal message path, starting with the current object.

As with any message, you can override the message path by using the send command to trigger a message handler in any object.

You call a function handler by using its name followed by parentheses (which enclose any parameters) in an expression. The function handler follows the same message-path rules as message handlers. These are examples of function calls:

```
put someFunction(firstParam,secondParam) into myVariable
put myFunction() into field "Help"
```

You can override the message path for a function call by using the value function to specify the object that the function handler resides in.

Function handlers use the return control structure to return a value to the calling handler.

Note: Transcript is not an òeverythingÆs a functionö language. If a routine does not return a value other than an error message, you normally implement it in a message handler rather than a function handler.

Parameters

Parameters can be passed to any handler, either by value or by reference. They are named in the first line of a handler, separated by commas. For example, a custom message handler that takes three parameters might look like this:

```
on myMessage firstParam,secondParam,thirdParam
  put firstParam into field 1
  put secondParam into field 2
  put thirdParam into field 3
end myMessage
```

In this example, òmyMessageö is the name of the handler (and therefore the name of the message that triggers it). òfirstParamö, òsecondParamö, and òthirdParamö are three parameters that may be passed to this handler. The handler can be called with a statement that looks like this:

```
myMessage "Some string",2,8+4
```

If a parameter value is not supplied by the calling statement, its value in the handler is the empty string.

Variables and sources of value

Transcript supports scalar and array variables, which can contain any kind and amount of data. You can also use function results, object contents, and object properties as sources of value in statements, interchangeably with variables.

Typelessness of variables:

Although arrays are supported, Transcript is otherwise a typeless language. The compiler performs appropriate casting internally when performing numeric operations, but from the viewpoint of a Transcript developer, all variables are treated as strings. It is not necessary to assign a type to a variable.

You can test the data type of the contents of a variable (or any other source of value) using the `is a` and `is not a` operators.

Declaring and assigning variables:

Variables can contain any amount of text or binary data. You assign a variable's value using the `put` command:

```
put "something or other" into myVariable
```

Local variables need not be declared or initialized. If a variable does not already exist, putting something into it declares it as a local variable and initializes it to the specified value.

If you wish, you can explicitly declare local variables with the `local` command. (You can set the `explicitVariables` global property to `true` to force all variables to be explicitly declared.) A variable that is declared but not initialized has the empty string as its value.

Scope of variables:

By default, variables are local, with scope being the currently executing handler. If you create a variable by putting something into it, the variable is a local variable.

If you explicitly declare a variable using the `local` command, the variable's scope depends on where the `local` command is placed. If it's inside a handler, the variable is local to that handler. If the `local` command is placed in a script, but outside any handler in that script, the variable is scoped to all the handlers in that script. Such variables are called script local variables.

To declare a global variable, you use the `global` command. The scope of a global variable is every handler in the application. However, to use a global variable in a handler, you must declare it, either in the handler or in the script the handler belongs to. Otherwise, Transcript assumes the variable is a local one. (Re-declaring a global variable doesn't re-create it, if it's already been created: it only makes it accessible to the handler or script.)

Array variables:

Transcript supports array variables. Arrays are associative, meaning you can use any value—not just integers—as the key for an array element. For example, the keys of an array might be `âä`, `âb`, and `âc`, or `âThis`, `âThat`, and `âTheOtherThing`:

put myArray["TheOtherThing"] into myTempThing

As with ordinary (scalar) variables, arrays can be local, script local, or global. Array variables are declared and assigned in the same way as scalar variables.

If an array already exists, you create a new element in it by putting something into the element. If you put something into an array that doesn't yet exist, Revolution creates the array as a local variable:

put "Hello" into myArray[1] -- creates myArray

Other sources of value:

In addition to variables, Transcript can use functions, object properties, parameters, field text, button contents, image contents, and URLs as sources of value.

In general, sources of value can be used interchangeably in any Transcript expression, without needing to read the source of value into a variable. For example, this expression—consisting of the sum of field text, a variable, and a function result—is a valid expression in Transcript:

(field "Some Number") + someNumericVar + currentSum()

Chunk expressions:

Transcript also supports a powerful method for accessing parts of a string, called chunk expressions. A chunk expression can be used to specify any line, word, or character in a string, or any item. Items, by default, are delimited by commas, but you can use the `itemDelimiter` property to change the delimiter. Chunk expressions can also specify a range of lines, words, characters, or items. You can specify one chunk type within another. Here are a few valid chunk expressions:

word 2 of myVariable
char 1 to 3 of field "Data"
line -4 of button "Menu" -- 4th-from-last line
item 2 of line 6 of URL "http://www.example.com/stuff.html"

Remember that all variables in Transcript are treated as strings, so you can use chunk expressions with any variable. See the topic [About chunk expressions](#) for more information about chunk expressions.

Control structures

Transcript supports the following standard control structures:

ò **Loops:** Use the repeat control structure to loop through a set of statements a specified number of times, while a specified condition is true, by iteration over a set of values, or until the loop is explicitly broken with an exit repeat statement.

ò **Conditionals:** Use the if control structure to conditionally execute a set of statements. Use the switch control structure to select from a number of cases.

ò **Exception handling:** Use the try control structure to execute a set of statements while handling errors internally.

Control structures can be nested to any depth.

Libraries and code re-use

The message hierarchy allows a single handler to control multiple objects. For example, the controls on a card are lower in the hierarchy than the card object, so a `mouseDown` handler in the card's script can respond to mouse clicks on any object that's on the card. Similarly, a handler in a stack script is available to every object in that stack. Strategic placement of handlers in the message hierarchy helps avoid duplication of code between similar objects.

To filter a handler so that it only responds to certain objects, rather than every object below it in the hierarchy, use the `target` function to determine which object originally received the message being handled.

The `insert script` command adds an arbitrary object to the message hierarchy. To create a code library, place the handlers you want to re-use in any object that's available in your stack, then use the `insert script` command to add that object to the hierarchy. Handlers in that object's script are now accessible to any other handler in Revolution.

What's Next?

Next, try the *Getting Started Tutorial*, which will give you a hands-on grounding in the process of building a simple application and writing scripts for it.

The *Independent Study Tutorial* should be your next stop. It provides a sample application and tells you how to study its scripts. The scripts in the sample application contain plentiful comments to explain what each line does.

As you continue working with Transcript, get into the habit of looking up terms in the Transcript Dictionary. The dictionary includes every Transcript command, function, operator, property, and control structure, with information on how you use each of them in a statement. Scanning the Dictionary can be very helpful, even if you don't remember everything you read, because it will give you an idea of what is possible with Transcript and where to look for a feature.

Tip: When you see a boldfaced Transcript term in the documentation, clicking it opens the Transcript Dictionary to the entry for that term.

Once you have your feet wet in Revolution, the best way to learn more is to build an application for yourself. Decide on a simple application to meet some of your own needs—a searchable address book, a digital clock, or whatever appeals to you—and write it from scratch. Or re-implement a simple application you already have and use—maybe one you've always wished worked a little differently or had an additional feature.

Summary

In this topic, you have learned that:

- Revolution applications are based on objects: each user-interface element is an object, whose appearance and behavior are determined by its properties. You can define custom properties for any object, which can hold any kind of data.

ò Stack objects are the top-level objects. Cards are contained in stacks. Controls (fields, buttons, images, graphics, scrollbars, players, and EPS objects) appear on cards, and can be collected into group objects. Audio clips and video clips are part of a stack and do not appear on a card.

ò Each object contains a script, which consists of routines called handlers. A handler can execute a command or respond to a message (message handler), evaluate a function (function handler), or respond to getting or changing a custom property (getProp handler or setProp handler).

ò Revolution programs work by means of messages. Revolution sends messages to objects in response to user actions (such as clicking or typing) and in response to system conditions. You can also send messages from within a script. The message path determines how messages travel from object to object. You can override the message path using the pass and send commands.

ò Transcript, Revolution's scripting language, is a very-high-level language, English-like and highly compressed. It includes control structures for conditionals (if and switch control structures), loops (repeat control structure), and exception handling (try control structure).

ò You create objects, edit code, test scripts, and create the final application all in the same development environment.

ò To learn more about Revolution, work through the tutorials, scan the Transcript Dictionary, and experiment with creating a simple application.

About messages and the message path

See Also:

About object types and object references, How to call a custom function thatÆs not in the message path, How to create a code library, How to include an object in the message path of every object, How to intercept a message, How to monitor messages as theyÆre sent, How to trap a message before a handler is executed, How to use a handler outside the message path, Development menu > Suppress Messages, mainStack property, me keyword, owner property, target function, target keyword

Transcript is based upon the sending of messages. Every action a script takes is ultimately triggered by a message being sent to that scriptÆs object.

This topic discusses how messages are sent, what happens to a message that is not handled by the object itÆs sent to, how to prevent messages from being sent, and how to change the usual message-passing behavior.

To fully understand this topic, you should know how to create objects and write short scripts. If you have gone through the “Getting Started” tutorial, you have enough information to fully understand this topic.

Contents:

- Messages and Message Sending
- The Object Hierarchy and the Message Path
- Trapping, Passing, and Sending Messages
- Extending the Message Path
- Summary

Messages and Message Sending

A message is an announcement that an event has occurred, which is made to the object that the event affects. When an event occurs, Revolution sends the affected object the appropriate message for the event.

What causes messages to be sent?

Events include user actions (such as typing a key or clicking the mouse button) and program actions (such as completing a file download or quitting the application). Revolution watches for events and sends a message to the appropriate object when an event occurs.

These messages are referred to as built-in messages, and include mouseDown, mouseUp, keyDown, openCard, and all the other messages described in the Transcript Dictionary.

Note: When a tool other than the Browse tool is active, the development environment traps the built-in messages that are normally sent when clicking (such as mouseDown and mouseUp). This is so that, for example, you can use the Pointer tool to select and move a button without triggering its mouseUp handler.

Revolution also sends a message whenever a handler executes a custom command. (Built-in commands are executed directly by the engine and donÆt result in sending a message.) When you execute a custom

command, Revolution sends a message, which has the same name as the command, to the object whose handler is currently executing. (For more information about custom commands, see the topic "About commands and functions".)

Similarly, Revolution sends a function call whenever a handler calls a custom function, a setProp trigger whenever a handler sets a custom property, and a getProp call whenever a handler gets the value of a custom property.

Responding to messages

To respond to a message, you write a handler with the same name as the message. For example, to respond to a keyDown message sent to a field (which is sent when the user presses a key while the insertion point is in the field), place a keyDown handler in the field's script:

```
on keyDown theKey -- responds to keyDown message
  if theKey is a number then beep
end keyDown
```

You respond to a custom command's message in the same way, by writing a message handler with the name of the command:

```
on myCommand -- a custom command
  beep 3
  go next card
  add 1 to field "Card Number"
end myCommand
```

Similarly, you write a function handler to respond to custom function calls, a setProp handler to respond to setProp triggers, or a getProp handler to respond to getProp calls.

Passing messages between objects

All messages are sent to a particular object. For example, when the user clicks an object, Revolution sends a mouseDown message to the object that was clicked. If the object's script contains a mouseDown handler, the message causes the handler to execute.

Function calls are sent to the object whose script the function is called in. (If you call a function from the message box, the call is sent to the current card.) setProp triggers and getProp calls are sent to the object whose custom property is being set or queried.

If the object's script doesn't include a handler for the message, it's passed on to another object to see whether that object can handle the message instead. The process by which a message is passed from one object to another until the message finds an appropriate handler is called the message path.

The Object Hierarchy and the Message Path

Each Revolution object is part of another object, of a different object type. For example, each card is part of a stack, each grouped control is part of a group, and so on. The object hierarchy defines these ownership relations between objects.

The object hierarchy is not the same as the path that messages go through, but it is closely related. In most cases, when an object passes a message on, the message goes to the object's owner in the object hierarchy.

The object hierarchy

All of Revolution's object types can be arranged in an object hierarchy, with each object being owned by one on the level above it, another object owning that one, and so on.

For example, a button's object hierarchy includes the button itself, the card the button is on, and the stack that the card is in, in that order. (For more information about the object hierarchy for each type of object, see the topic "About object types and object references".)

The message path

The message path is the set of rules that determine which objects, in which order, have the opportunity to respond to a message. The message path is based on the object hierarchy.

Revolution sends each message to a particular object. If the object's script contains a handler for that particular message, the handler is executed. If the object's script can't handle the message, it's passed on to the next object in the message path. This process continues until the message either finds an object that can handle it, or reaches the end of the message path.

For example, suppose the user clicks a button in a main stack, causing Revolution to send a mouseUp message to the button:

1. If the button's script does not contain a handler for the mouseUp message, the message is passed along to the card the button is on.
2. If the card's script contains a mouseUp handler, the handler is executed. But if the card does not handle the mouseUp message, it is passed on to the card's stack.
3. If the stack script contains a mouseUp handler, the handler is executed. But if the stack does not handle the mouseUp message, it is passed on to the engine. The engine is the end of the message path, and if a mouseUp message reaches it, the engine throws the message away. If neither the original button nor any of the other objects in the message path has a mouseUp handler, then the mouseUp message doesn't do anything.

Message path rules:

If an object's script has no handler for a message, the message is passed to the next object. These are the rules that determine which object is next:

ò Card controls > card

The next object is the card the control is on.

ò Grouped controls > group

The next object is the group the control is in.

ò Groups > card or stack

If the group is on the current card, the next object is the current card. If the group is not on the current card, the next object is the first card the group is on. If the group is not placed on any card, the next object is the stack the group is in.

(In some cases, a card can receive a message before a group. If the card has already received the message, that message isn't passed to the card again.)

ò Cards > stack or group

If the card has no groups whose `backgroundBehavior` property is set to true, the next object is the stack the card is in.

If there are any such groups on the card that haven't received the message yet, the next object is the first one of these groups. The message then goes to the rest of the `backgroundBehavior` groups on the card, in order by number. After all the `backgroundBehavior` groups have received the message, it is sent to the stack.

ò Substacks > main stack

The next object is the substack's main stack.

ò Main stacks > standalone

The next object is the application's main stack (if the stack is running in a standalone application).

ò Standalone > engine

The last object in the message path is the engine.

An object receives a message only once. If any object in the message path handles the message (and the handler doesn't use the pass control structure to let the message keep going), the message stops at that object and goes no further in the message path.

Custom function calls, `setProp` triggers, and `getProp` calls go through the message path in the same way as messages.

Note: If a stack's `dynamicPaths` property is set to true, handlers in that stack use HyperCard's dynamic path behavior: if a handler uses the `go` or `find` command to go to a card other than the original card, that destination card's message path is inserted into the message path as long as the handler is on that card. The `dynamicPaths` property is provided for compatibility with imported HyperCard stacks, and is normally set to false, but you may encounter this behavior when working with a stack that was originally created in HyperCard.

The message target:

The object that a message was originally sent to is called the message's target. You can get the target from within any handler in the message path by using the `target` function.

The target is usually the same as the object whose script is being executed, but not necessarily. For example, if you click a button (causing a `mouseUp` message to be sent), and the button's script contains a `mouseUp` handler, then the `target` function returns the button's name. However, if the button doesn't handle the `mouseUp` message, it's passed to the card, and if the card has a `mouseUp` handler, it is executed in response to the message. In this case, the card's script is executing, but the target is not the

cardùitÆs the button that was originally clicked, because Revolution sent the mouseUp message to the button.

Tip: To get the name of the object whose script is currently executing, use the me keyword.

Two handlers with the same name:

If two different objects in the message path each have a handler with the same name, the message is handled by the first object that receives it and has a handler for it.

For example, suppose that a buttonÆs script includes a mouseUp handler, and so does the stack script. If you click the button, a mouseUp message is sent to the button. Because the buttonÆs script has a mouseUp handler, the button handles the message, and it isnÆt passed any further. The message is never sent to the stack script, so for this click event, the stack scriptÆs mouseUp handler is not executed.

Note: If an objectÆs script has more than one handler with the same name, the first one is executed when the object receives the corresponding message. Other handlers in the same objectÆs script with the same name are never executed.

Unhandled messages:

If a built-in message, a setProp trigger, or a getProp call passes through the entire message path without finding a handler, it is ignored by the engine (the last stop in the message path).

If a message corresponding to a custom command or a custom function call reaches the end of the message path without finding a handler, it causes an execution error.

Trapping, Passing, and Sending Messages

When an object receives a message and a handler for that message is found, the handler is executed.

Normally, a message thatÆs been handled does not go any further along the message path. Its purpose having been served, it disappears. Such a message is said to have been trapped by the handler.

Trapping a message

If you want to prevent a message from being passed further along the message path, but donÆt want to do anything with it, an empty handler for the message will block it from being passed on. This example prevents the mouseDown message from being acted on by the object the empty handler is in, as well as any objects below it in the object hierarchy:

```
on mouseDown
end mouseDown
```

You can use the same technique to block custom function calls, setProp triggers, and getProp calls.

Blocking navigation messages:

You can block navigation messagesùmessages related to moving between cardsùfrom being sent while a handler is executing by setting the lockMessages property to true.

For example, if the handler opens another stack, Revolution normally sends openCard and openStack messages to the stack. If the stack contains handlers for these messages that would cause unwanted behavior during this operation, you can use the lock messages command before going to the stack in

order to temporarily block these messages. When the handler finishes executing, the `lockMessages` property is automatically reset to its default value of `false`, and normal sending of messages resumes.

Tip: To block navigation messages while testing or debugging a stack, choose Development menu Suppress Messages to place a checkmark next to the option.

Passing a message to the next object

To let a message pass further along the message path, use the `pass` control structure. The `pass` control structure stops the current handler and sends the message on to the next object in the message path, just as though the object hadn't contained a handler for the message:

```
on openCard
  doSpecialSetupForThisCard
  pass openCard -- let stack get the message too
end openCard
```

Usually, you should allow built-in messages to pass by placing a `pass` command at the end of your handler.

Selectively trapping or passing messages:

Some built-in messages, such as `keyDown`, trigger an action, so trapping the message prevents the action from being performed at all. The following example passes the `keyDown` message only if the character typed is a number:

```
on keyDown theKey
  if theKey is a number then pass keyDown
end keyDown
```

If you place this handler in a `fieldÆs` script, it allows the user to enter numbers, but any other keystrokes are trapped by the `keyDown` handler and not allowed to pass.

A similar principle applies to `setProp` triggers. If a `setProp` handler does not pass the `setProp` trigger, the custom property is not set.

Sending a message

If the handler you want to use is not in the message path, you can use the `send` command to send the corresponding message to the object whose script contains the handler.

For example, suppose you have a handler for a custom command called `myCommand` in the script of a button, and you want to execute that handler from a card script. Since the button is not in the `cardÆs` message path, you can't simply use the command in the script, because if the `myCommand` message goes through the `cardÆs` message path, it won't find the handler. The following statement sends the message directly to the button whose script includes the handler:

```
send "myCommand" to button "Stuff" of card "Stuff Card"
```

You can also use the `send` command without specifying an object:

```
send "myCommand"
```

If you don't specify an object, the message is sent to the object whose script is being executed. Using the send command without specifying an object is equivalent to simply executing the custom command:

```
myCommand
```

The only difference is that if you use the send command without an object, any parameters of the command are evaluated before the message is sent.

The message path of a sent message:

Messages you send also proceed through the target object's message path, if the object doesn't trap the message. For example, if you send a message to a card button, and the button script doesn't have a handler for that message, the message is passed to the card the button is on, then to the stack, and so on, following the rules given above for the message path.

Important! When you send a message to an object, the message path for that message starts with the target object. For example, if a stack script contains a send command that sends a message to a button, the message moves through the button's message path, not the stack's.

Because messages that are sent by the send command proceed along the message path, the same as messages sent by Revolution, you can use this technique to skip a portion of the message path. For example, suppose you want a mouseUp message that's received by a button to be passed directly to the stack, and to skip the card. The following example traps the original mouseUp message (so it doesn't go on to the card), and sends a new mouseUp message directly to the stack:

```
on mouseUp -- in button script
    send "mouseUp" to this stack
end mouseUp
```

The send command versus the call command:

The call command is similar to the send command. Like the send command, the call command sends a message to the specified object, allowing you to use handlers that aren't in the message path.

The difference between the two is in how they handle object references in the handler that is triggered by the sent message. If the message is sent by the send command, object references in the handler are treated relative to the object you sent the message to.

For example, suppose card 1 of a stack contains the following handler:

```
on showCard
    answer the number of this card
end showCard
```

If a handler in the script of card 3 uses the send command to send a `showCard` message to card 1, the dialog box displays `1`, the number of the card the handler is on. However, if the same handler uses the call command instead, the dialog box displays `3`, the number of the card that used the call command. In other words, handlers that are triggered by the send command use the context of the object the handler is in, while handlers that are triggered by the call command use the context of the object that triggered the handler.

Sending a function call:

If you want to use a custom function whose function handler is not in the message path, you can use the value function to specify the object. The value function can be thought of as an equivalent of the send command, for function calls instead of messages.

For example, if card 1 of a stack contains a function handler called `myFunction`, you can use the function from within the script of card 3 with the following statement:

```
get value("myFunction(1,2,3)",card 1)
```

Extending the Message Path

What if the message path isn't enough? For example, you might want to be able to easily use a set of utility handlers from any script in your application. You could send the required message to the object containing the utility handlers, but this is a little cumbersome, and may cause problems if you later shift those handlers into another object's script.

Instead, you can extend the message path by adding objects to it. Using the insert script and start using commands, you can add an object to the message path of every object in your application, so the added object's script is available to every handler.

Creating a code library with backscripts

You can extend the message path by adding objects to it. To create a code library, place the handlers you want to re-use in any object that's available in your stack, then use the insert script command to add that object to the message path:

```
insert script of card "Library" into back
```

Handlers in that object's script are now accessible to any other handler in Revolution.

The script of an object that's been inserted into the back, as in the above example, is called a backscript. Such an object is placed last in the message path of all objects. It receives messages after any other object, and just before the engine receives the message. The backscript object remains in the message path until the end of the session (or until you remove it with the remove script command.)

Because a backscript receives messages after all other objects in the message path, if you place a handler in the backscript, it eventually gets all the corresponding messages, no matter what object sent them, unless another object handles them first.

Using a stack's script with start using:

The start using command is similar to the insert script command, but can be used only to place stacks, not other object types, in the message path.

The start using command inserts the stack at the end of the message path, after the object's stack and main stack, but before objects that have been inserted into the back with insert script.

Trapping messages with frontscripts

You can also extend the message path by placing objects into the front of the message path, before any other object. The script of such an object is called a frontscript, and you use the insert script command to place the object in the front:

insert script of button "Interceptor" into front

Because the object is in the front of the message path, it receives messages even before the target of the message. For example, if you click a button, any objects in the front of the message path receive the mouseUp message first, before the button. If you place a handler in a frontscript, it receives all the corresponding messages before any other object can handle them.

Use a frontscript when you want to be able to handle a message even if the target object has a handler for it. For example, the Revolution development environment displays a contextual menu when you Command-Control-Shift-click or Control-Shift-right-click an object. It does this with a mouseDown handler in a frontscript. Whenever you click an object, the frontscript receives the mouseDown message first, and checks whether the needed keys are being pressed. If they are, the handler displays the contextual menu; the mouseDown message is trapped and goes no further. Otherwise, the handler passes the message to the next object in the message path, which is the object you clicked.

Summary

In this topic, you have learned that:

- ò A message is an announcement that an event has occurred. Revolution sends messages to objects in response to user actions and system events, and you can also send your own messages in a handler.

- ò A message handler is executed when the object whose script contains the handler receives a message with the same name as the handler.

- ò If the object that receives a message doesn't have a handler for it, the message is sent to the next object in the message path. The message path is the set of rules that governs which objects receive a message, and in what order.

- ò You can stop a message at any point in the message path with an empty handler for the message. You can use the pass control structure to pass a message to the next object after you're done with it.

- ò The send command can send any message to any object, with or without a time delay.

- ò To insert additional objects into the message path of every object, you use the insert script and start using commands.

About the structure of a script

See Also:

About commands and functions, About custom properties and custom property sets, Memory and Limits Reference, How to create a code library, How to edit an object's script, How to include comments and notes in a script, How to temporarily remove a portion of a script, Recipe for listing all the handlers in a script, end keyword, function control structure, getProp control structure, local command, on control structure, script property, setProp control structure

A script is an object property containing all the Transcript code for that object. This topic discusses the elements that can appear in a script, the overall structure of a script, and the process of compiling scripts.

To fully understand this topic, you should know how to create objects and write short scripts. If you have gone through the "Getting Started" tutorial, you have enough information to fully understand this topic.

Contents:

What is a Script?

Declarations and Comments

Handlers

Compiling a Script

Summary

What is a Script?

A script is a collection of structured text stored in a property, called the script property. Each Revolution object has a script, although the script may be empty. You edit the script's text using the script editor.

When Revolution compiles an object's script property, it makes a list of the script's handlers. Later, when Revolution sends a message to an object, it checks to see whether the object has a handler for the message, and executes the handler if so. From this description, you can see that it's possible to think of a script as a collection of handlers, all contained in the script property, one after another.

In addition to handlers, scripts can contain comments (remarks that aren't in Transcript and aren't executed) and declarations for certain types of variables and constants.

Like any property, the script property can be set in a handler or the message box with the set command. When you use the Revolution script editor to make changes to an object's script, the Apply button sets the object's script property to the new contents of the script.

Declarations and Comments

Declarations and comments are the parts of a script that can be placed either in a handler, or outside any handler.

Comments

Any line (or portion of a line) that starts with two dashes (--) or a hash mark (#) is a comment. Comments are ignored by Revolution, even if they contain Transcript statements. Placing these characters at the beginning of a line is called "commenting out" the line.


```
on mouseUp -- this part is a comment
    beep
    -- and this is a comment too
end mouseUp
```

You can temporarily remove a statement, or even an entire handler, by commenting it out. (To comment out several lines at once, select them and choose Script menu Comment in the script editor.)

Since comments are not executed, you can place them anywhere in a script—inside a handler or outside all handlers.

Block Comments:

Comments that start with -- or # only extend to the end of the line. To create a multiple-line comment, surround it with /* and */:

```
on openCard
    /* This is a multiple-line comment that might
    contain extensive information about this handler,
    such as the author's name, a description, or the
    date the handler was last changed. */
    show image "My Image"
    pass openCard /* You can also make one-line comments */
end openCard
```

Block comments are handy when you want to temporarily remove a section of code while debugging a handler. You can place the characters /* at the start of the section, and */ at the end, to prevent the section from being executed.

Declarations

A declaration is a command that creates a variable or constant. Declarations made with the global, local, or constant commands can be placed either inside or outside a handler. If a declaration appears in a handler, the variable or constant is available only to that handler. If the declaration appears outside any handler, the variable or constant is available to all the handlers in the script.

Declarations are the only commands that can be placed outside any handler.

Handlers

Each handler is a complete section of code, and can be executed independently. There are four types of handler: message handlers, function handlers, getProp handlers, and setProp handlers.

Message handlers

Each message handler begins with the on control structure followed by the name of the message that this handler responds to. The handler ends with the end control structure and the name of the message.

Message handlers look like this:

```
on mouseUp
    beep
end mouseUp
```

A message handler is executed when the object whose script contains the handler receives the message. This example handler responds to the mouseUp message.

You can write handlers for any built-in message. You can also write a handler for your own custom message, and send the message to an object either by using the send command or by using the name of the message as a command.

Function handlers

Each function handler begins with the function control structure followed by the name of the function that this handler computes. The handler ends with the end control structure and the name of the function. Function handlers look like this:

```
function currentDay
  return item 1 of the long date
end currentDay
```

A function handler is executed when a handler in the same script (or one in an object lower in the object hierarchy) calls the function. This example handler returns todayÆs name.

getProp handlers

Each getProp handler begins with the getProp control structure followed by the name of the custom property that this handler corresponds to. The handler ends with the end control structure and the name of the property. getProp handlers look like this:

```
getProp myCustomProperty
  return the scroll of me + 20
end myCustomProperty
```

A getProp handler is executed whenever the value of the corresponding custom property is requested by a Transcript statement.

You can write a getProp handler for any custom property of the object or another object lower in the object hierarchy.

setProp handlers

Each setProp handler begins with the setProp control structure followed by the name of the custom property that this handler corresponds to. The handler ends with the end control structure and the name of the property. setProp handlers look like this:

```
setProp myCustomProperty newSetting
  set the hilite of me to true
  pass setProp
end myCustomProperty
```

A setProp handler is executed whenever the value of the corresponding custom property is changed by the set command.

You can write a setProp handler for any custom property of the object or another object lower in the object hierarchy.

For more information about message handlers and function handlers, see the topic [About commands and functions](#). For more information about getProp and setProp handlers, see the topic [About properties, custom properties, and property sets](#).

All handlers, regardless of their type, must start with one of the four control structures and end with the end control structure. The message, function, or property name specified in the first line must match the name in the last line.

Compiling a Script

A script is compiled when you change the script either by using the set command or by clicking Apply in the script editor. During compilation, the entire script is analyzed.

You cannot change a script while a handler in it is executing, because what is executed is the compiled version, not the text in the script property.

If a compile error is found when a script is being compiled, the entire script is unavailable for execution until the error is corrected and the script is re-compiled. (This applies only to compile errors. If an execution error occurs during the execution of a handler, it does not affect the ability to use other handlers in the same script.)

Summary

In this topic, you have learned that:

- ò Each object has a script, which can be empty or can contain one or more Transcript handlers. You change a script using the script editor, or by setting the object's script property.
- ò A comment is a part of the script that is not executed. Comments start with -- or #.
- ò A script can contain four kinds of handlers: message handlers, function handlers, setProp handlers, and getProp handlers.
- ò Declarations (using the global, local, and constant commands) are the only Transcript code that can be used outside a handler.
- ò If a script contains a compile error, none of its handlers can be used until the error is fixed.

About commands and functions

See Also:

About the structure of a script, Memory and Limits Reference, How to create a code library, How to create a synonym for a command or function, How to execute a command, How to find out whether a command or function is defined, How to return an array from a function, How to return multiple values from a handler, How to write your own commands and functions, Recipe for listing all the handlers in a script, commandNames function, function control structure, functionNames function, on control structure, return control structure

Commands and functions are the basic building blocks of a Transcript statement. Commands perform an action, while functions compute a value. In addition to using Transcript's built-in commands and functions, you can write your own commands and functions, which can be used by other handlers.

This topic discusses how commands and functions are used, how to write handlers that implement custom commands and functions, and how to pass information between the calling handler and the custom message handler or function handler.

To fully understand this topic, you should know how to create objects and write short scripts that use Transcript functions and commands. If you have gone through the "Getting Started" tutorial, you have enough information to fully understand this topic.

Contents:

- Using Commands and Functions

- Creating Custom Commands and Functions

- Passing Parameters

- Returning Values

- Summary

Using Commands and Functions

You use commands and functions to perform the action of your application. Commands instruct the application to do something—such as play a movie, display a window, or change a property. Functions compute a value—different functions might add a column of numbers, or get the fifteenth line of a certain file, or find out whether a key is being pressed.

Built-in commands and functions

Transcript has over a hundred fifty built-in commands, and over two hundred built-in functions, all of which are documented in the Transcript Dictionary.

Using commands in a statement:

A command is an instruction to Revolution to do something. Every program statement starts with a command, and every line of Transcript code—other than control structures such as if and repeat—is a statement.

Here are some examples of how built-in commands are used in statements:

```
go next card -- go command
```

beep -- beep command
set the hilite of me to true -- set command

Using function calls in a statement:

A function call is a request to Revolution for information. When you use a function in a statement, Revolution calls the function to get the specified information, then substitutes that information for the function call.

Here's an example of how a function is used:

```
put round(22.3) into field "Number"
```

When this statement is executed, Revolution calls the round function. When you round off 22.3, the resulting number is 22, so the statement puts the number 22 into the field.

The number you're rounding off is placed in parentheses after the round function's name. This number is called the function's parameter. A function can have one parameter, or none, or several. The parameters are enclosed in parentheses and, if there's more than one, they're separated with commas. Here are some examples:

```
put date() into myVariable -- date function, no parameters  
put length("hello") into me -- length function, 1 parameter  
get average(10,2,4) -- average function, 3 parameters
```

Important! A function call, by itself, is not a complete statement. You need to use a function call in conjunction with a command or control structure. (In the first example above, the round function is used with the put command.)

Writing function calls with the `fieldName` form:

If a built-in function has no parameters or one parameter, it can be written in a different form, with no parentheses:

```
put the date into myVariable -- date function  
put the length of "hello" into me -- length function
```

If the function has no parameters, this form is written as the `fieldName`. If it has one parameter, this form is written as the `fieldName of parameter`.

The `fieldName` form works the same way as the `functionName()` form shown above, and you can use the two forms interchangeably for built-in functions with fewer than two parameters. (The Transcript Dictionary entry for each built-in function shows how to write both forms.)

Important! You can use the `fieldName` form for built-in functions, but not for custom functions that you write. (Writing custom functions is discussed later in this topic.)

Library commands and functions

In addition to its set of built-in commands and functions, Revolution includes a variety of custom libraries: the Animation library, Database library, Internet library, Printing library, Profile library, Speech library, Video library, XML library, and Common library.

Like built-in commands and functions, custom library commands and functions are described in the Transcript Dictionary. The name of the library is displayed above the command or function name.

For the most part, the commands and functions in these libraries work the same way as built-in commands and functions. You can use them in statements in any handler. Here are some examples of library commands and functions in use:

```
revPlayAnimation "BouncingBall"  
revSpeak "Hello, how are you?"  
put revAppVersion() into field "Version"  
get libURLErrorData("http://www.example.com")
```

A note about syntax differences:

Built-in commands can have very flexible syntax:

```
go to card 3 of stack "Dialogs" as modal  
group image "Help Icon" and field "Help Text"  
hide button ID 22 with visual effect dissolve very fast
```

However, the syntax of library commands is more limited. A library command can several parameters, and if there is more than one, they are separated by commas:

```
libURLDownloadToFile myFile,newPath,"downloadComplete"  
revExecuteSQL myDatabaseID,field "Query","*b" & "myvar"
```

But library commands cannot use words like *and* and *to* to join parts of the command together, the way built-in commands can. Because of this, library commands cannot be as English-like as built-in commands can be.

Similarly, library functions don't have an alternate *the* form, and always use the form with parentheses, even if they have one parameter or no **Parameters**:

```
put revAppVersion() into theVersion  
get revNumberOfRecords(myDatabase)
```

Custom commands and functions

Although Revolution has many built-in commands and functions, you can also create your own custom commands and custom functions, or use custom commands and functions created by other Revolution developers.

Note: To find out how to make custom commands and custom functions, see *Creating Custom Commands and Functions* below.

You use custom commands and custom functions the same way as any other command or function.

Using custom commands in a statement:

Like a built-in command or library command, a custom command is an instruction to Revolution to do something. You use custom commands in statements, as in the following made-up examples:

```
grabThumbnail
checkForConnection "ftp://ftp.example.org"
makePrefsFile fileLoc,field "Preferences"
```

Note: Custom commands have the same syntax limitations as library commands do.

Using custom functions in a statement:

Like a built-in function or library function, a custom function call is a request for information. Here are a few examples showing the use of made-up custom functions:

```
get formattedPath("/Disk/Folder/File.txt")
put summaryText(field "Input") into field "Summary"
if handlerNames(myScript,"getProp") is empty then beep
```

Note: Like library functions, custom functions don't have a `do` form, and are always written with parentheses following the function name. If the function has parameters, they are placed inside the parentheses, separated by commas. If the function doesn't have parameters, the parentheses are empty.

Creating Custom Commands and Functions

A custom command is a command whose behavior is defined in a message handler that you write. You can include any statements you want in this handler, so you can create a custom command to perform any set of tasks that Revolution is capable of.

A custom function, like a custom command, is defined by a handler you write (a function handler, in this case). A custom function can include any statements you want, so you can create a custom function to perform any computation or get any information that Revolution can access.

The basics of custom commands and functions

The anatomies of message handlers and function handlers differ only slightly. Message handlers begin with the `on` control structure, while function handlers begin with the `function` control structure. Next comes the name of the message or function call that the handler responds to, plus a list of any parameters.

This is an example of a message handler for a custom command called `getTotals`:

```
on getTotals
  put field 1 + field 3 into field "Subtotal"
  if field "Subtotal" > zero then beep
end getTotals
```

When you use the custom command `getTotals`, the handler is executed.

This example shows a function handler for a custom function called `fileHeader`, with one parameter:

```
function fileHeader theFile
  put char 1 to 8 of URL ("file:" & theFile) into tempVar
  put binaryDecode("h*",tempVar) into tempVar
  return tempVar
```

end fileHeader

When you use the custom function `fileHeader` in a statement, the function handler is executed, and the function call is replaced by the value in the return statement. This example shows how the function can be used:

put fileHeader(it) into myFileHeaderVar

Why create commands and functions?

Writing custom commands and functions offers several benefits:

ò Re-use code:

If you have a set of statements that you use often, you can put those statements in a separate handler. Then, in your other handlers, you can replace your set of statements with the custom command or function call. As well as saving space, this tactic makes it easier to change the functionality of your code, since you only need to change it once—in the separate handler—instead of in many handlers.

ò Make your code more readable:

Because a custom command or function call is a single line, it's easier to make sense of than a whole series of statements. By using custom commands and functions, you can break up a handler into smaller, more manageable pieces.

ò Create code libraries:

A set of custom commands and custom functions can be placed in a single object—a code library—and you can copy this library to any application of yours that requires them. Having a set of standardized, debugged commands and functions for basic or common tasks saves you development time.

Should it be a command or a function?

As you can see from the examples above, custom command handlers are very similar to custom function handlers. Both can execute multiple statements, you can pass parameters to either, and you can return values from either. Since they're so similar, you might be wondering when to create a custom function and when to create a custom command.

In general, you should create a custom command if the handler's main purpose is to perform a task. If the handler's main purpose is to return a value to the handler that called it, create a custom function instead.

Custom commands

You create a custom command by writing a message handler with the same name as the command, and placing the handler in the script of an appropriate object. The handler is executed when the object receives a message with the same name as the handler.

Custom command messages:

Revolution sends a message whenever a handler executes a custom command. (Built-in commands are executed directly by the engine and don't result in sending a message.) When you execute a custom command, Revolution sends a message, which has the same name as the command, to the object whose handler is currently executing. (For more information about messages, see the topic `About messages and the message path`.)

If the object's script contains a message handler with the same name as the command, that message handler is executed. Otherwise, the message is passed along the message path until it finds a handler with the same name.

A custom command handler:

To create a custom command, you create a message handler with the name of your command, and place it in the script of the object where it's going to be used, or else in the script of an object that's further in the message path. When the command is executed, Revolution sends the message, triggering your message handler. In that handler, you can perform any task you want to attach to your custom command.

For example, suppose you want to alert the user by sounding a beep and displaying a dialog box. Create a simple message handler and place it in your stack's script:

```
on alertUser theMessage
    beep
    answer theMessage
end alertUser
```

Now, you can include an `oalertUser` command in any script in your stack. When Revolution encounters the `oalertUser` custom command in a handler, it sends an `oalertUser` message through the message path, triggering the handler above.

For example, the following handler beeps and displays the dialog box:

```
on mouseUp
    alertUser "You clicked a button!"
end mouseUp
```

Function calls and custom functions

Creating a custom function is very similar, except that you create a function handler rather than a message handler. This custom function places spaces between each character of a string:

```
function expanded theString
    repeat for each character nextChar in theString
        put nextChar & space after expandedString
    end repeat
    return char 1 to -2 of expandedString
end expanded
```

To use this custom function in another handler, include the function call as part of an expression. Here is an example using the above custom function:

```
on mouseUp
    ask "What string do you want to expand?"
    answer expanded(it)
end mouseUp
```

When a handler calls a function that's not one of Revolution's built-in functions, Revolution sends the function call to the object whose script contains the handler. If the script contains a function handler

with the same name as the function being called, that function handler is executed. Otherwise, the function call is passed along the message path until it finds a handler with the same name. (For more information about messages, see the topic "About messages and the message path".)

Creating a code library

A library is a set of custom commands and custom functions for a specific application or a specific area of functionality. You can create a library for any purpose you want, and put any custom commands and functions into it that you need. You can also exchange useful libraries with other developers.

To create a code library, place the handlers you want to use in any object that's available in your stack. This object is now a code library. Then use the insert script command to add that object to the message path:

```
insert script of card "Library" into back
```

Handlers in that object's script are now accessible to any other handler in Revolution, which means you can use the library object's custom commands and custom functions in any script.

If you make a stack into a code library, you can easily exchange the library with other developers simply by passing around the stack file.

Passing Parameters

A value that you pass from one handler to another is called a parameter.

In the "alertUser" example above, the following statement sends the "alertUser" message with a parameter:

```
alertUser "You clicked a button!"
```

The parameter "You clicked a button!" is passed to the "alertUser" handler, which accepts the parameter and places it in a parameter variable called "theMessage". The "alertUser" handler can then use the parameter in its statements:

```
on alertUser theMessage
  beep
  answer theMessage -- uses the parameter "theMessage"
end alertUser
```

Passing multiple parameters

If a statement passes more than one parameter, the parameters are separated by commas. The following example has two parameters, "theMessage" and "numberOfBeeps":

```
on seriouslyBugUser theMessage,numberOfBeeps
  beep numberOfBeeps
  answer theMessage
end seriouslyBugUser
```

To use this custom command, you call it like this:

```
seriouslyBugUser "Hello",5
```

When the `seriouslyBugUser` handler is executed with the statement above, the `theMessage` parameter is `"Hello"`, and the `numberOfBeeps` parameter is 5.

Parameter variables

In the example above, `theMessage` and `numberOfBeeps` are the parameter variables. You declare parameter variables in the first line of a handler. When the handler begins executing, the values you passed are placed in the parameter variables. You can use parameter variables the same way as ordinary variables: you can put data into them, use them in expressions, and so on.

Parameter variables are local variables, so they go out of existence as soon as the handler stops executing.

(Formally, the value you pass to the handler is a parameter, and the special variable in the handler is called a parameter variable, but both these concepts are often referred to with the word `parameter` for simplicity.)

Parameter variable names:

You can give a parameter any name that's a legal variable name. (The names of variables must consist of a single word and may contain any combination of letters, digits, and underscores (`_`). The first character must be either a letter or an underscore.)

It is not the name, but the order of parameters that is significant.

Empty parameters

A handler can have either more or fewer parameter variables than the number of parameters passed to it. If there are more parameter variables than there are values to put in them, the remaining parameter variables are left empty. Consider the following handler, which has three parameter variables:

```
on processOrder itemName,itemSize,numberOfItems
  put itemName into field "Name"
  put itemSize into field "Size"
  if numberOfItems is empty then put 1 into field "Number"
  else put numberOfItems into field "Number"
end processOrder
```

The following statement places an order for one sweater:

```
processOrder "sweater","large"
```

The statement only passes two parameters, while the `processOrder` handler has three parameter variables, so the third parameter variable, `numberOfItems`, is empty. Because the handler provides for the possibility that `numberOfItems` is empty, you can pass either two or three parameters to this handler.

Setting a default value for a parameter:

To use a default value for a parameter, you check whether the parameter is empty. If it is, then no value has been passed, and you can simply put the desired default into the parameter, as in the following example:

```
logData theData,theFilePath
  if theFilePath is empty then
    put "logfile" into theFilePath
  end if
  put theData into URL ("file:" & theFilePath)
end logData
```

The `logData` handler puts data into a file, whose name and location you specify in the second parameter. If you only provide one parameter, the handler uses the filename `logfile` as the default value, and logs the data to that file:

```
logData field 1,"/Disk/Folder/data.txt" -- specifies a file
logData myVariable -- doesn't specify a file, uses "logfile"
```

The first statement above specifies the second parameter, so it doesn't use the default value. The second statement only specifies one parameter, so the data will be placed in `logfile` by default.

Implicit parameters

If a statement passes more parameters than the receiving handler has parameter variables to hold them, the receiving handler can access the extra parameters with the `param` function:

```
function product firstFactor,secondFactor
  put firstFactor * secondFactor into theProduct
  if the paramCount > 2 then
    repeat with x = 3 to the paramCount
      multiply theProduct by param(x)
    end repeat
  end if
  return theProduct
end product
```

The function above assumes that two parameters will be passed to be multiplied, but can multiply more numbers by using the `param` function to access parameters beyond the first two. The following statement uses the `product` custom function above to multiply four numbers together:

```
answer product(22,10,3,7)
```

When the `product` handler executes, the first two parameters—22 and 10—are placed in the parameter variables `firstFactor` and `secondFactor`. The third parameter, 3, is accessed with the expression `param(3)`, and the fourth parameter, 7, is accessed with the expression `param(4)`.

Passing parameters by reference

Normally, if you pass a variable name as a parameter, that variable is not changed by anything the called handler does. This is because the variable itself is not passed, only its contents. Passing parameters in

this way is called "passing by value" because the variable's value—not the variable itself—is what is passed.

If the name of a parameter variable is preceded with the @ character, that parameter's value is a variable name, rather than the value in the variable. Changing the parameter variable in the called handler changes the value of the variable in the calling handler. This way of passing parameters is called "passing by reference", because you pass a reference to the variable itself instead of just its value.

For example, the following handler takes a parameter and adds 1 to it:

```
on setVariable @incomingVar
  add 1 to incomingVar
end setVariable
```

The following handler calls the "setVariable" handler above:

```
on mouseUp
  put 8 into someVariable
  setVariable someVariable -- call by reference
  answer "someVariable is now:" && someVariable
end mouseUp
```

Executing this mouseUp handler displays a dialog box that says "someVariable is now: 9". This is because, since "someVariable" was passed by reference to the "setVariable" handler, its value was changed when "setVariable" added 1 to the corresponding parameter variable.

You can pass parameters by reference to any custom function or custom command, simply by preceding the parameter name with the @ character in the first line of the handler, as in the "setVariable" example handler above. (Do not use the @ character when referring to the parameter elsewhere the handler.)

Note: If a parameter is passed by reference, you can pass only variable names for that parameter. You cannot pass string literals or expressions using other containers such as fields. Trying to use the "setVariable" command described above using the following parameters will cause an execution error:

```
setVariable 2 -- can't pass a literal by reference
setVariable field 2 -- can't pass a container
setVariable line 1 of someVariable -- can't pass a chunk
```

Empty **Parameters:**

If a handler defines a parameter as being passed by reference, you must include that parameter when calling the handler. Omitting it will cause an execution error.

Returning Values

Once a function handler has calculated a value, it needs a way to send the result back to the handler that called the function. And if an error occurs during a message handler, it needs a way to send an error message back to the calling handler.

In Transcript, you use the return control structure to return a value from a handler to the handler that called it.

Returning a value from a function handler

The return control structure is used within a function handler to pass the resulting value back to the calling handler. The returned value is substituted for the function call in the calling statement, just like the value of a built-in function. Take another look at the example from above:

```
function expanded theString
  repeat for each character nextChar in theString
    put nextChar & space after expandedString
  end repeat
  return char 1 to -2 of expandedString
end expanded
```

In the custom function example above, the return control structure sends the spaced-out string back to the mouseUp handler that called the `expanded` function.

Note: The return control structure stops the handler, so it's usually the last line in the handler.

Returning an error from a message handler

When used in a message handler, the return control structure serves a slightly different purpose: it returns an error message to the calling handler.

When used in a message handler, the return control structure sets the result function for the calling handler. If you want to return an error message to the calling handler, use the return control structure within the message handler.

Here's an example of a message handler that displays a dialog box:

```
on echoAMessage
  ask "What do you want to show?"
  if it is empty then return "No message!"
  else answer it
end echoAMessage
```

This handler asks the user to enter a message, then displays that message in a dialog box. If the user doesn't enter anything (or clicks Cancel), the handler sends an error message back to the calling handler. A handler that uses the `echoAMessage` custom command can check the result function to see whether the command successfully displayed a message:

```
on mouseUp
  echoAMessage
  if the result is empty then beep
end mouseUp
```

Note: The result function is also set by many built-in commands in case of an error. If you check the result in a handler, the value belongs to whatever command—built-in or custom—you set it last, so if you're going to check the result, be sure to do so right after the command whose success you want to check.

Summary

In this topic, you have learned that:

ò A command instructs the application to do something, while a function requests the application to compute a value.

ò You create a custom command or custom function by writing a handler for it.

ò Values that you pass to a handler are called parameters.

ò To pass a parameter by reference, you precede its name with an @ sign in the first line of the handler.

ò When used in a function handler, the return control structure returns a value.

ò When used in a message handler, the return control structure returns an error message that can be accessed with the result function.

About containers, variables, and sources of value

See Also:

About chunk expressions, About custom properties and custom property sets, About properties and property profiles, About using URLs, Image Types Reference, Memory and Limits Reference, Operator Precedence Reference, How to display the contents of a text file, How to find out whether a container is an array, How to include a comma in a parameter, How to include a quote in an expression, How to monitor the value of variables while debugging, How to put text into a field, How to search a container, How to store styled text in a variable or property, Why does a variable lose its value?, Recipe for 99 Bottles of Beer on the Wall (using send), [] keyword, () operator, constant command, delete variable command, get command, global command, local command, put command, variableNames function

A source of value is any way of referring to data that you can use in a statement. Sources of value include variables, function results, arithmetic expressions, and literal strings. A container is a source of value that you can also put data into, such as a field or variable.

This topic discusses the different sources of value you can use in a Transcript statement, how to use variables to hold a value, and how to use other containers, including objects and URLs.

To fully understand this topic, you should know how to create objects and write short scripts. You should also have a basic understanding of what a handler is, what a parameter is, and what a variable is. If you have gone through the *Getting Started* tutorial, you have enough information to fully understand this topic.

- Contents:
- Using Containers
- Variables
- Other Containers
- Other Sources of Value
- Compound Values and Operators
- Summary

Using Containers

A container holds data that you can change and retrieve. Transcript uses several different kinds of containers: variables, URLs, and objects like fields that can hold data.

One way to think of a container in a concrete way is to view it as a sort of box, which can hold information. You can add things to the box, remove things, replace the box's contents, and find out what's in the box.

Seeing what's inside a container

To retrieve the contents of a container, you either use it in an expression, or use the get command. Here are some examples that show how the content of one kind of container, a variable, can be used:

```
get myVariable
add 10 to myVariable
answer myVariable + 20 -- "myVariable + 20" is an expression
```



```
put myVariable into field "My Field"
```

The last example shows the use of two containers: it gets the content of the variable `myVariable`, and puts it into a field.

Changing the content of a container

To put something into a container, you use the `put` command:

```
put "Hello" into field "My Field"
put 1+2 into theVariable
put field "Data" into URL "file:data.txt"
```

When you use the `put` command in this way, it replaces whatever used to be in the container with the new contents. You can also use the `before` and `after` keywords to put data into a container without disturbing whatever is already in it:

```
put space & "World" after field "My Field"
put "3=" before theVariable
```

To remove a container's content without deleting the container itself, you put empty into the container:

```
put empty into field "My Field"
put empty into theVariable
```

Portions of a container (chunk expressions)

You're not limited to dealing with the whole container at once. To specify a portion of what's in a container—a line, an item, a word, or a character—you use a chunk expression:

```
get line 2 of myVariable
put "123" before item 7 of field "My Field"
put return after word 25 of theVariable
```

You can also specify ranges, and even one chunk inside another:

```
put char 1 to 20 of field "My Field" into myVar
answer word 3 of line 5 of theVariable
```

You can use any chunk of a container in the same way you use a whole container: put data into the chunk (or before or after it) and find out what data it contains. In this way, any chunk of a container is like a container itself.

For more information about chunk expressions, see the topic [About chunk expressions](#).

Variables

A variable is a container you create, which has no on-screen representation, but which you can refer to in a handler or the message box. Variables can hold any data you want to put into them, and their contents can be retrieved at any time.

One way to think of a variable is as a box with a name on it. You can put anything you want into the box, and take it out later, by simply providing the variable's name:

```
put 1 into thisThing -- a variable named "thisThing"
put thisThing into field ID 234
put "Hello Again!" into line 2 of thisThing
```

But unlike some containers, variables are non-permanent and aren't saved with the stack. Instead, variables are automatically deleted either when their handler is finished running or when you quit the application (depending on the variable's scope). You can also use the delete variable command to delete a variable. When a variable is deleted, not only the content of the variable disappears, but also the variable itself—the box.

Tip: To save a variable's value, in your application's closeStackRequest or shutdown handler, set a custom property of the stack to the value of the variable. To restore the variable, in your application's startup or openStack handler, put the custom property into the variable.

Variable scope

The scope of a variable is the part of the application where the variable can be used. If you refer to a variable in a handler that's outside the variable's scope, you'll get either an execution error or an unexpected result.

There are three levels of variable scope: local, script local, and global. Every variable is one of these three types. The difference between these types is in where they can be used and how long their value lasts.

Local variables:

A local variable can be used only in the handler that creates it. Once the handler finishes executing, the variable is deleted. The next time you execute the handler, the variable starts from scratch: it does not retain what you put into it the last time the handler was executed.

To create a local variable, you either declare it with the local command, or simply put something into it. If you use the put command with a variable name that does not yet exist, the variable is automatically created as a local variable:

```
put true into myNewVar -- creates variable named "myNewVar"
```

You can create a local variable explicitly by using the local command inside a handler:

```
local myNewVar -- creates variable named "myNewVar"
put true into myNewVar -- puts a value into "myNewVar"
```

Important! If you use a local variable in one handler, and that handler calls another handler, you can't use the local variable in the second handler. (If you use a variable with the same name, Revolution creates a second variable that is local to the second handler. But the two local variables don't affect each other, because they're in different handlers.)

In the following example, the two handlers each have a local variable named `myVar`, but they are different local variables because they're in different handlers, and changing one does not affect the other:

```
on mouseUp
  put 1 into myVar -- creates a local variable
  doCalledHandler
  answer myVar -- displays "1", not "2"
end mouseUp

on doCalledHandler
  put 2 into myVar
  -- creates a different variable with the same name
end doCalledHandler
```

Local variables are deleted when the handler that they're used in finishes executing. You can also use the `delete variable` command to delete a local variable.

Script local variables:

A script local variable can be used in any handler in an object's script. You can use a script local variable in any handler in the same script, but not in handlers in other objects' scripts. Unlike a local variable, a script local variable retains its value even after a handler finishes executing.

To create a script local variable, you use the `local` command in the script, but outside any handler:

```
local mySharedVariable

on mouseDown
  put 2 into mySharedVariable
end mouseDown

on mouseUp
  answer mySharedVariable -- displays "2"
end mouseUp
```

Any handler that comes after the local declaration can use the script local variable. It is customary to place all local declarations for script local variables at the top of the script. This ensures that every handler in the script can use the variable. It also makes the declarations easy for you to find.

Note: If you put the `local` command in a handler, instead of outside any handler, it creates a local variable instead. The command creates a script local variable only if you put it in the script but not within a handler.

Script local variables are automatically deleted when the application quits or when the script they're declared in is recompiled. (A script is recompiled when you click **Apply** in the script editor, when you close the script editor after changing the script, and when you use the `set` command to change the script.) You can also use the `delete variable` command to delete a script local variable.

Global variables:

A global variable can be used in any handler, anywhere in the application. Unlike a local variable, a global variable retains its value even after the handler that created it finishes executing. Unlike a script local variable, a global variable can be used by any handler in any object's script.

The same global variable can be used by any stack during a session. You can declare a global variable in one stack, and use it in others.

To create a global variable, you use the global command:

```
global someGlobalSetting
```

You also use the global command to make an existing global variable available to a handler. While a global variable can be used by any handler, you need to use the global command to make it available. If you don't declare a global variable before using it, the handler will not take the global variable's existence into account, and will simply create a local variable with the same name.

You can use the global command either inside a handler, or outside any handler. If you use the command in a handler, the global variable can be used by any statement that comes after it in that handler. If you use the command in a handler but outside any script, the global variable is available to every handler that comes after the global declaration.

The following example shows the use of a global variable in a button script. In this example, the variable is declared outside any handler, so the individual handlers don't need to declare it again:

```
global myGlobal -- declares this global for the whole script
```

```
on mouseDown -- can use "myGlobal"  
  put 1 into myGlobal  
end mouseDown
```

```
on mouseUp -- can use "myGlobal"  
  add 2 to myGlobal  
  answer myGlobal -- displays "3"  
end mouseUp
```

To use the same global variable in a handler where the variable isn't declared in the script, you must place the global declaration in the handler:

```
on mouseUp -- in another button's script  
  global myGlobal  
  add 5 to myGlobal  
  answer myGlobal  
end mouseUp
```

If you click the first button, then the second, the second button displays the number 8.

It is customary to place all global declarations in scripts at the top of the script, and to place all global declarations in handlers at the top of their handlers. This ensures that every handler in the script (or every statement in the handler) can use the variable. It also makes the declarations easy for you to find.

Global variables are automatically deleted when you quit the application. You can also use the delete variable command to delete a global variable.

Tip: You can get a list of existing global variables with the `globalNames` function. You can also choose Development menu Variable Watcher to see a list of global variables and change their values.

Variable names

The names of variables must consist of a single word and may contain any combination of letters, digits, and underscores (`_`). The first character must be either a letter or an underscore.

Here are some examples of legal variable names:

```
someVariable
picture3
my_new_file
_output
```

Here are some names that cannot be used as variable names:

```
3rdRock    -- starts with a digit
this&That  -- "&" cannot be used
My Variable -- more than one word
```

Avoid giving a variable the same name as a custom property. If you refer to a custom property, and there is a variable by the same name, Revolution uses the contents of the variable as the name of the custom property. Generally this will produce unexpected results.

Note: Global variables whose names begin with `ôgRevô` are reserved by the Revolution development environment.

Declaration and initialization

Declaring a variable is the process of either creating a new variable or making an existing variable available. (You use the same commands for both actions.) Initialization is the process of assigning a starting value to a variable.

Explicit declaration:

You declare local variables and script local variables with the `local` command, and declare global variables with the `global` command.

If the `local` command is used in a handler, the variable created is a local variable. If the command is used outside a handler, the variable is a script local variable.

The `global` command can be used either inside a handler or outside all handlers in a script. If the `global` command is used in a handler, the global variable can be used in that handler. If it is used outside a handler, the global variable can be used by all handlers in the script.

You need to declare all global and script local variables. Declaring local variables is optional.

Implicit declaration of local variables:

As mentioned above, local variables do not need to be declared: if you put something into a variable that doesn't exist yet, the variable is created automatically. This is called implicit declaration, because the variable declaration is implicit in using it.

There is no such thing as implicit declaration of global variables or script local variables. Using a non-existent variable creates it as a local variable.

The `explicitVariables` property:

Although using the `local` command is not necessary to create a local variable, you may prefer to declare local variables explicitly if it helps you keep better track of them.

One common source of bugs involves misspelling a local variable name. Normally, doing so doesn't produce an execution error, because if you use a variable that doesn't exist, Revolution creates it automatically. This means that if you misspell a variable name, Revolution creates a new variable with the misspelled name. Such a bug may be difficult to track down because it can result in a variable having the wrong value without causing an error message.

To prevent this problem, you can require all local variables to be declared with the `local` command. You do this by setting the `explicitVariables` property to true. If the `explicitVariables` is true, trying to use a local variable that doesn't exist will cause an execution error, instead of automatically creating it. Any misspelled variable names will therefore cause an obvious execution error when their handler is executed, making them easy to find.

Important! Setting the `explicitVariables` to true while the development environment is running may produce unexpected results. To safely test whether a stack's local variables are all explicitly declared, choose Development menu Suspend Development Tools before setting the `explicitVariables` property to true.

Initializing local variables:

With the `local` command, you can optionally set the variable to its initial value:

```
local someLocalVar = 3
```

If you do not specify an initial value with the `local` command, the variable is created empty.

It's not possible to initialize global variables and script local variables. Instead, they are created empty, and you can put whatever you want in them.

Special variable types

Most of the time, you use variables that you create yourself (using the `local` or `global` commands, or simply by putting a value into a new variable in order to create it).

Transcript also creates certain types of variables automatically: parameter variables, command-line variables, environment variables, and the special variable `it`.

Parameter variables:

In a handler for a custom command or custom function, you can define parameters on the first line of the handler. For example, the following handler defines two parameters named `thisThing` and `thatThing`:

```
on myHandler thisThing,thatThing
  add thisThing to thatThing
  subtract thatThing from field 2
end myHandler
```

When you use the custom command or custom function, you can pass values to it using the **Parameters**:

```
myHandler 15,4+1
-- puts "15" into the parameter "thisThing",
-- and puts "5" into the parameter "thatThing"
```

When named parameters are used in a handler, they are called parameter variables. Within the handler, the parameters can be used in the same way as local variables: you can get their value, use them in expressions, and put data into them.

Like local variables, parameter variables persist only as long as the handler is executing.

Environment variables:

On Unix and OS X systems, the operating system provides information about the operating environment in environment variables.

You can access environment variables by prepending the `$` character to the environment variable's name. For example, the following statement gets the contents of the `LOGNAME` environment variable, which holds the current user's login name:

```
get $LOGNAME
```

(See your operating system's technical documentation to find out what environment variables are available.)

You can also create your own environment variables by prepending the `$` character to the new environment variable's name:

```
put field 3 into $MYENVVAR
```

The environment variables that you create this way are available to the application, and are also exported to processes started up by the shell function or the open process command.

Environment variables behave like global variables and can be used in any handler. You do not need to use the global command to declare them before using them.

Command-line argument variables:

If you start up the application from the command line (on OS X, Unix or Windows systems), the command name is stored in the variable `$0` and any arguments passed on the command line are stored in numbered variables starting with the `$` character.

For example, if you start the application by typing the following shell command:

```
myrevapp -h name
```

then the variable \$0 contains `myrevapp` (the name of the application), \$1 contains `-h`, and \$2 contains `name`.

Command-line argument variables behave like global variables and can be used in any handler. You do not need to use the `global` command to declare them before using them.

The special variable `it`:

The `it` variable is a special local variable used by Revolution to store certain results.

Certain commands—such as `get`, `convert`, `read from file`, `ask`, and `answer`—put their results in this special variable. (For a complete list of commands that use the `it` variable, see the entry for `it` in the Transcript Dictionary.) The following example shows how the `answer` command uses the `it` variable:

```
on mouseUp
  answer "Go where?" with "Backward" or "Forward"
  -- the answer command puts the button the user clicked
  -- into the it variable:
  if it is "Backward" then go back
  else go next
end mouseUp
```

You can use the `it` variable in the same way as any other local variable, using the `put` command to put things into it and using the variable in expressions to work with the contents.

Array variables

A variable can hold more than a single value. A variable that holds more than one value is called an array, and each of the values it holds is called an element. Each element has its own name (called the element's key).

If you think of a variable as a box with a name, you can think of an array as a box with compartments inside it. Each compartment is an element, and each compartment has a name, or key, of its own.

You specify an element of an array variable by using the variable name along with the element's key. You enclose the key in square brackets.

Here's an example that shows how to put data into one element of an array:

```
put "ABC" into myVariable["myKeyName"]
```

Note: If you use a key that's not a number, you should enclose the key's name in double quotes whenever you refer to it. This prevents problems in case there is a variable or reserved word with the same name.

Arrays can be used much like chunk expressions. You can use any element of an array variable in the same way you use the whole variable: put data into the element (or before or after it) and find out what data it contains. In this way, any element of a variable is like a container itself.

For more information about array variables, see the topic [About arrays and matrixes](#).

Other Containers

In addition to variables, Revolution has six other container types: fields, buttons, images, URLs, the selection, and the message box. You can put information into all these containers with the `put` command, and use the container's name in an expression to get its contents.

Fields, buttons, and images are all Revolution objects. All display their content on the screen, in different ways, and the contents of all three are saved when you save the stack they are in. URLs refer to external resources (either files on the system, or items on an Internet server). The message box is a special container that's part of the development environment.

Fields

A field is a container that is part of a stack. Fields are represented on the screen as rectangular boxes filled with text; you can use a field's properties to change its appearance or even hide it completely.

Unlike variables, fields are part of a stack. Their content is stored when you save the stack they're in, so fields don't disappear when the application quits, the way variables do.

Note: To access or change a field's contents, the stack that includes the field must be open or be loaded into memory.

Using the content of fields:

You can use the content of a field by using a field reference in any expression. Transcript substitutes the field's content for the field reference:

```
put field 1 into myVar -- puts field content into variable
```

Like other containers, fields can be referred to by name:

```
put 1 into field "My Field"  
put field "Your Field" into myVariable
```

Because fields are objects, you can also refer to them by number or by ID:

```
get field 12  
put field "Info" into field ID 1734
```

And because fields are objects, if a field is on a card other than the current card, or in a stack other than the current stack, you need to specify the field's location when naming it:

```
put it after field "Help" of card 2
```

(For more information about how to refer to fields, see the topic [About object types and object references](#).)

Changing a field's content:

A field's content consists of its visible text. If the field's `lockText` property is set to false and its `traversalOn` is set to true, the user can edit the text. Otherwise, the field is locked against editing. (You can use the `put` command to change the field's contents, even if it's locked.)

As with other containers, you can change the whole field at once or change a part of it using a chunk expression. You can also use the `before` and `after` keywords. The following examples show how to change a field's contents using the `put` command:

```
put "ABC" into field "My Field" -- replaces any contents
get char 1 of field "My Field" -- yields "A"
put "Z" into char 2 of field "My Field" -- replaces "B"
put "DEF" after field "My Field" -- appends to end
get field "My Field" -- yields "AZCDEF"
put empty into field "My Field" -- empties it out
```

Note: Fields can hold any text data you want to put in them, but putting non-printable characters in a field may have unexpected results, so fields should not be used to store binary data.

Using the text property:

The content of a field is the same as its text property. You can change the contents by either putting the content into the field, or setting its text property. The following two statements are equivalent:

```
put "ABC" into field "Holder"
set the text of field "Holder" to "ABC"
```

You can get the content of a field by either using a field reference in an expression, or getting the field's text property:

```
get the text of field "Holder" -- yields "ABC"
get field "Holder" -- yields "ABC"
```

However, you cannot change a chunk of a field using the text property.

Styled text in fields:

Unlike other containers, fields hold not just text, but text styles. Text displayed in a field can include different fonts, different text sizes, different text and background colors, and styles such as bold and italic.

However, when you refer to the field as a container, the style, size, and font of text is not retained. For example, the following command does not retain any formatting of text in the first field:

```
put field 1 into field 2
```

To move text between fields (or chunks of fields) while retaining style information, use the `htmlText` property:

```
set the htmlText of field 2 to the htmlText of field 1
```

Buttons

A button, like a field, is a container that is part of a stack and is saved with the stack.

The content of a button is not displayed in the button. Instead, if the button is a button menu, the content is used as the list of menu items. If the button is not a menu (that is, if its style property is not `ômenuö`), its content is not displayed on screen, although you can still access and change the content.

Note: To access or change a button's content, the stack that includes the button must be open or be loaded into memory.

Button content and button menus:

The content of a button is displayed in whatever menu type is specified by the button's `menuMode` property. To create the individual menu items that will appear in the menu, you put the menu items into the button, one menu item per line. (You can either do this in a handler, or fill in the box labeled `ôMenu itemsö` on the Basic Properties pane of the property inspector.)

If the `menuMode` is `ôpulldownö`, `ôpopupö`, `ôcomboboxö`, or `ôoptionö`, the button's content is displayed as the list of menu items, one line per menu item. (For example, if a button's content includes three lines of text, its menu has three menu items.) If the `menuMode` is `ôtabbedö`, there are as many tabs as there are lines in the button, and each line in the button is displayed as the text of a tab.

Certain special characters affect the way the menu is displayed. For example, if a line contains `ô(ö`, the corresponding menu item is disabled. If a line begins with a tab character, the corresponding menu item is part of a cascading menu. Other special characters create keyboard shortcuts or indicate that a menu item should be marked in a special way. For more information about the use of special characters to affect menu display, see the topic `ôAbout menus and the menu barö`.

If a button is empty, no menu appears for it, even if its style is `ômenuö`.

Using the content of buttons:

You use the content of a button by using a button reference in any expression. Transcript substitutes the button's content for the button reference:

```
put button "Popup" into field "Menu Items List"
```

Like other containers, buttons can be referred to by name:

```
put button "My Button" after thisVar  
put button "Your Button" into myVariable
```

Because buttons are objects, you can also refer to them by number or by ID:

```
get button 6  
put field "Help" into button ID 1734
```

And because buttons are objects, if a button is on a card other than the current card, or in a stack other than the current stack, you need to specify the button's location when naming it:

put it & return after button "Choices" of card "Index"

(For more information about how to refer to buttons, see the topic [About object types and object references](#).)

Changing a button's content:

As with other containers, you can change the whole button at once or change a part of it using a chunk expression. You can also use the before and after keywords. The following examples show how to change a button's contents using the put command:

```
put "ABC" into button "My Button" -- replaces any contents
get char 1 of button "My Button" -- yields "A"
put "Z" into char 2 of button "My Button" -- replaces "B"
put "DEF" after button "My Button" -- appends to end
get button "My Button" -- yields "AZCDEF"
put empty into button "My Button" -- empties it out
```

Note: As with fields, buttons can hold any text data you want to put in them, but putting non-printable characters (other than whitespace characters) in a button may have unexpected results when the button menu is displayed.

Using the text property:

The content of a button is the same as its text property. You can change the contents by either putting the content into the button, or setting its text property. The following two statements are equivalent:

```
put "Go" into button "My Menu"
set the text of button "My Menu" to "Go"
```

You can get the content of a button by either using a button reference in an expression, or getting the button's text property:

```
get the text of button "My Menu" -- yields "Go"
get button "My Menu" -- yields "Go"
```

However, you cannot change a chunk of a button using the text property.

Images

Images, like fields and buttons, are containers that are part of a stack and are saved with the stack. Unlike fields and buttons, however, images hold binary data instead of text.

The content of an image is the raw binary data that makes up the picture in an image. If you understand the format that the image is in, you can analyze or alter the image by using the put command to change portions of the data in the image.

The content of a referenced image is empty, because the picture is stored in a separate file instead of in the image object.

Note: To access or change an image's content, the stack that includes the image must be open or be loaded into memory.

Using the content of images:

You can use the content of an image by using an image reference in any expression. Transcript substitutes the image's content for the image reference:

```
put image 1 into myVar -- puts image content into variable
```

Like other containers, images can be referred to by name:

```
put myVar into image "My Image"  
put image "Your Image" into anotherVariable
```

Because images are objects, you can also refer to them by number or by ID:

```
get image 3  
put image "Graph" into image ID 1243
```

And because images are objects, if an image is on a card other than the current card, or in a stack other than the current stack, you need to specify the image's location when naming it:

```
put lastPix after image "My Picture" of first card  
get image 23 of card "My Card" of stack "My Stack"
```

(For more information about how to refer to images, see the topic "About object types and object references".)

Image formats and the content of an image:

The content of an image consists of the raw image data, in whatever format the image is stored in. Revolution supports GIF, JPEG, PNG, PICT (on Mac OS and OS X only), BMP, PBM/PGM/PPM, and XBM/XPM/XWD formats, as well as its own internal format. You can find out what format an image is in by checking the image's paintCompression property.

Image content is binary data, not text. This means that the content may contain any characters, including control characters, so you cannot simply display the data in a field. (Putting the content of an image into a field typically results in displaying nonsense characters.) To meaningfully work with the content of an image, you must know what format the image is in, and understand the details of that format.

Important! Images stored in Revolution's internal format—whose paintCompression property is 0—do not include all the image data. This is the only image format for which this is the case.

Changing an image's content:

Because the content of an image is not text, it does not make sense to talk about lines, words, or items of an image, generally speaking. However, because each byte of the image content corresponds to a character, you can work with the image content using chunk expressions.

For example, to display the value of a particular byte of the image's content, use the charToNum function:

```
put charToNum(char 12 of image "My Image")
```

into field "Display" -- shows numeric value of 12th byte

To translate a numeric value into a byte of data, use the numToChar function:

```
put numToChar(244) into char 20 of image "My Image"
```

As with other containers, you can change the whole image at once or change a part of it using a chunk expression. You can also use the before and after keywords.

To copy the content of one image to another, use the put command:

```
put image "Original" into image "Working Copy"
```

To remove the picture from an image without deleting the image object itself, use a statement like the following:

```
put empty into image "Working Copy"
```

Using the text property:

The content of an image is the same as its text property. You can change the contents by either putting the content into the image, or setting its text property. The following two statements are equivalent:

```
put myVariable into image "Holder"  
set the text of image "Holder" to myVariable
```

You can get the content of an image by either using an image reference in an expression, or getting the image's text property:

```
get the text of image "Holder" -- yields data  
get image "Holder" -- yields same data
```

However, you cannot change a chunk of an image using the text property.

Note: Despite the name of this property, the content of an image is binary data, not text.

Image content and the imageData property:

The imageData property of an image holds binary data that makes up the image. This property is related to the content of the image—changing either one changes what's displayed in the image—but they're not identical. The imageData property and the image content are in different forms, have different sizes, and include overlapping but not identical information about the picture:

ò The content of an image is the raw data that makes up the picture, in the image's own format (PNG, JPEG, GIF, PICT, or the internal RLE format). The format of the content, therefore, depends on the value of the image's paintCompression property.

The imageData, on the other hand, consists of the picture data presented in a standard form. The form of the imageData property, unlike the content, does not depend on what format the image is recorded in; it's always in the same form, which specifies the color of each pixel in the image, four bytes per pixel.

ò Because most picture formats include compression, the content of an image is normally smaller than its imageData property.

ò Because the imageData property records only the colors of pixels, it does not include the transparency mask or alpha channel, if the image has one. Instead, this information is found in the maskData and alphaData properties of the image. If an image includes transparency information, you need all three properties to fully describe the picture's appearance.

ò The imageData property specifies the appearance of the picture as it's presented on the screen, not stored in the image object. If you resize or rotate an image, the content of the image does not change, but its imageData does.

ò The content of a referenced image is empty, but the imageData of a referenced image is not. This is (again) because the imageData property reflects the on-screen appearance of the picture, rather than what's stored in the image object.

The imageData property and the image content, because of their different characteristics, are useful for different purposes. To manipulate each pixel (for example, to brighten the image or remove the red channel), to examine the actual screen appearance of the image, or to work with the picture data in a format-independent way, use the imageData property. To take advantage of the characteristics of a particular format or to transfer picture data between objects, use the image content.

URLs

A URL is a container for a file (or other resource), which may be on the same system the application is running on, or may be on another system that's accessible via the Internet. A URL can hold text or binary data in any format.

URLs, like fields, buttons, and images and unlike variables, are persistent: they do not disappear when you quit the application. Unlike fields, buttons, and images, URLs are not objects and are not part of a stack.

For more information about URLs, see the topic "About using URLs, uploading, and downloading".

URL schemes:

A URL scheme is a type of URL. Revolution supports five URL schemes: http, ftp, file, binfile, and (on Mac OS and OS X) resfile.

The http and ftp schemes designate data that is located on another system that's accessible via the Internet. The file, binfile, and resfile schemes designate local files.

Using the content of a URL:

You designate a URL with the URL keyword:

```
get URL "file:/Disk/Folder/file.txt"  
put URL "http://www.example.com/stuff" into field "Stuff"
```

The URL keyword tells Revolution that you are using the URL as a container. As with other containers, you use the content of a URL by using a reference to the URL in an expression. Transcript substitutes the URLÆs content for the reference.

If the URL scheme refers to a local file (file, binfile, or resfile URLs), Revolution reads the content of the file and substitutes it for the URL reference in the expression:

```
answer URL "file:../My File" -- displays the file's content
put URL "binfile:flowers.jpg" into myVariable
put URL "resfile:Icons" into URL "resfile:New Icons"
```

If the URL scheme refers to a file or resource on another system (http or ftp URLs), Revolution downloads the URL automatically, substituting the downloaded data for the URL reference:

```
answer URL "http://www.example.net/files/greeting.txt"
```

Note: The content of an ftp or http URL consists of whatever data is sent back when Revolution tries to download the URL. If the server sends back an error message—for example, if the file you specify in an http URL doesnÆt exist—then the error message replaces the URL reference in the expression.

Changing the content of a local URL:

Whether and how you change the content of a URL depends on the URL scheme.

For URLs that refer to a local file (file, binfile, and resfile URLs), you can change the whole URL by putting something into it, or change a part of the URL using a chunk expression. You can also use the before and after keywords. The following examples show how to change a file URL:

```
put "Hello" into URL "file:test.txt"
get char 1 of URL "file:test.txt" -- yields "H"
put space & "You" after URL "file:test.txt"
put "World" into word 2 of URL "file:test.txt"
get URL "file:test.txt" -- yields "Hello World"
put empty into URL "file:test.txt" -- empties the file
```

If a file or binfile URL doesnÆt already exist, putting something into the URL creates the specified file.

Because the content of a binfile or resfile URL is not text, it does not make sense to talk about lines, words, or items of such a URL, generally speaking. To get the value of a particular byte of a binfile URL, use the charToNum function:

```
answer charToNum(char 1 of URL "binfile:picture.jpg")
-- displays numeric value of 1st byte
```

To translate a numeric value into a byte of data, use the numToChar function:

```
put numToChar(20) into char 3 of URL "binfile:picture.jpg"
```

Changing the content of an http or ftp URL:

When you put something into an http or ftp URL, Revolution uploads the data to the URL.

What happens next depends on the server. Most HTTP servers don't allow uploading files, so putting something into an http URL usually results in an error message, instead of a successful upload. Most FTP servers require a valid user name and password, so putting something into an ftp URL that doesn't include this information usually results in an error message.

Because it's impossible to upload part of a file, you cannot put something into a chunk of an http or ftp URL. (You can get a chunk of such a URL. However, Revolution must download the entire file to find the chunk you specify.)

The message box

The message box is a special container. The expression message box refers to the lower field in the message box window, where the result of calculations is placed.

The message box is the default destination for the put command: if you don't specify what to put data, the put command places it in the message box. For example, the following statement puts "Hello World" into the message box:

```
put "Hello World"
```

The message box can hold only text data.

Important! Unlike other containers, the message box is present only in the development environment, not in standalone applications.

The selection

The selection is a reference to the currently selected text in a field. The selection container exists only if there is a text selection. The selection is always associated with text: objects and parts of an image can be selected, but they cannot be the selection.

If no text is selected, but there is an insertion point, the selection is the insertion point and the selection container is empty.

Using the content of the selection:

You can use the content of the selection by using the expression the selection in any expression. Transcript substitutes the selected text for the reference:

```
put the selection into myVariable
-- puts selected text into variable
```

As with other containers, you can use the entire selection, or get a part of it using a chunk expression:

```
put word 2 to 3 of the selection after field "Backup"
```

If the selection is empty (that is, if there is an insertion point but no text is selected), or if there is no selection, the expression the selection evaluates to empty.

Changing the selection:

If there is some text selected, putting text into the selection replaces the selected text with the new text.

If there is an insertion point but no text is selected, putting text into the selection inserts it at the insertion point.

If there is no selection, attempting to put something into the selection causes an execution error.

As with other containers, you can change the whole selection at once or change a part of it using a chunk expression. You can also use the before and after keywords. The following examples show how to change the content of the selection using the put command:

```
put "ABC DEF" into the selection -- replaces selection
put "Hello" into word 1 of the selection
put "World" after the selection
put "There" before word 2 of the selection
put empty into the selection -- removes the text
```

When you add text to the selection or replace the selection, the insertion point is placed at the end of the added text. When you remove the contents of the selection by putting empty into it, the insertion point is placed where the selection used to be.

The select command:

In addition to clicking and dragging to select text, you can use the select command to select text or place the insertion point in a field:

```
select text of field "My Field" -- selects entire field
select after word 2 of field 3 -- places insertion point
select char 4 to 7 of field ID 9345 -- selects some text
```

Other Sources of Value

In addition to containers, Transcript statements can use other sources of value: properties, function calls, literal strings, and constants. These sources of value, along with containers, are the building blocks from which all expressions are created.

Like containers, these sources of value can be used in any expression:

```
put the accentColor after field "Settings" -- property
put the date && the time into myTimeVariable -- functions
if "1 + 1" is field 1 then beep -- literal string
```

They are not containers, however, because you cannot use the put command to change them.

Properties

A property is an attribute of an object. You obtain a property's value by using its name along with a reference to its object:

```
put the showBorder of button 1 into thisBorder
```

Global properties are not associated with any object. You refer to the value of a global property by prefixing it with the word `the`:

```
get the accentColor  
put the idleTicks into myTicksVar
```

To change a property, you use the set command.

Tip: To see a list of all built-in properties, open the Documentation window, click Transcript Dictionary, and choose "Properties" from the menu at the top of the window.

For more information about properties, see the topic "About properties, custom properties, and property sets".

Function calls

A function call is a request to Revolution for information. When you use a function in a statement, Revolution calls the function to get the specified information, then substitutes that information for the function call.

A function call consists of the function name, plus any parameters the function requires, enclosed in parentheses:

```
put myFunction(2,5) after myVariable
```

Built-in functions with one parameter or no parameters can also be written in a different form, with no parentheses:

```
put the date into myVariable  
put the length of field "My Field" into me
```

If the function has no parameters, this form is written as the functionName. If it has one parameter, this form is written as the functionName of parameter.

Tip: To see a list of all built-in functions, open the Documentation window, click Transcript Dictionary, and choose "Functions" from the menu at the top of the window.

For more information about functions and function calls, see the topic "About commands and functions".

Literal strings

A literal string is a string of characters whose value is itself. If the string is a number, the value is that number. When you use a literal string in an expression, Transcript simply substitutes the string for itself:

```
put "Hello World!" into field 1  
get 1 + 2 + it  
put 1 - 4.234 into field "Result"
```

Quoting strings:

Literal strings that consist of more than one word or are reserved words in the Transcript language must be enclosed in double quotes:

```
put "This is a test" into myVar -- works
put This is a test into myVar -- DOESN'T WORK - not quoted
put That into myVar -- works
put This into myVar -- DOESN'T WORK - reserved word
```

In some contexts, you can use an unquoted one-word literal string without causing a script error. However, you should make a practice of always quoting literal strings (other than numbers), because it ensures that the statement will continue to work properly even if the string becomes a reserved Transcript word in the future.

Note: If the `explicitVariables` property is set to true, compiling a script that contains an unquoted literal string causes a script error.

Numbers and base conversions:

If an arithmetic expression is quoted, it is used as-is:

```
put "1+1" into field "Result" -- displays "1+1"
```

If the expression is not enclosed in double quotes, it's evaluated before being used:

```
put 1+1 into field "Result" -- displays "2"
```

If the first two characters of a number are 0x, the number is interpreted as a hexadecimal number when arithmetic is done on it:

```
answer 0xA + 0xB -- displays "21"
```

If the `convertOctals` property is set to true and the first character in a number is 0, the number is interpreted as an octal number.

Constants

A constant is a value that has a name. Like a variable, a constant is set in your script. Unlike variables, constants cannot be changed.

When you use a constant, Transcript substitutes the value of the constant for its name. The following example uses a constant named `slash`:

```
put slash after field "Expressions" -- displays "/"
```

You create a new constant using the `constant` command.

You cannot put anything into a constant once it's been created.

Built-in constants:

The Transcript language defines several constants, such as `return`, `space`, and `comma`, for characters that have special meaning in scripts and therefore can't be entered literally into an expression.

Tip: To see a list of all built-in constants, open the Documentation window, click Transcript Dictionary, and choose `Constants` from the menu at the top of the window.

User-defined constants:

You can also define your own constants using the constant command:

```
constant myName="Joe Smith"
```

Like variables, constants can have different scope depending on how they are created. A constant can be defined as either a local constant or a script local constant:

ò If you place the constant command in a handler, the constant can be used only in that handler.

ò If you place the constant command in a script, but outside any handler, the constant can be used in any handler in the script.

Compound Values and Operators

To create a compound expression, you use operators to combine any of the sources of value described above. Here are some examples:

```
put "1+2 =" && 1+2 into field "Eq" -- displays "1+2 = 3"  
put button "Menu" & return & newItem into button "New Menu"  
filter field "Data" with "[" & mySelection & "]"*"  
if the platform is "MacOS" and field "Time" < zero then...
```

Tip: To see a list of all operators, open the Documentation window, click Transcript Dictionary, and choose ôOperatorsö from the menu at the top of the window.

Types of operators

Operators include numeric operators such as + and -, string operators such as &, and logical operators such as and and there is a. The type of an operator determines what kind of value it produces.

For details about each operator, see its entry in the Transcript Dictionary.

Numeric operators:

Numeric operators produce a number as their result. Numeric operators include the arithmetic operators (+, -, *, /, mod, div, and ^) and the bitwise operators (bitAnd, bitOr, bitXOr, and bitNot).

String operators:

String operators produce a string of characters as their result. String operators are the concatenation operators (&, &&, and ,).

Logical operators:

Logical operators produce either ôtrueö or ôfalseö as their result.

Logical operators include the comparison operators (=, <>, <, >, <=, >=), existence operators (there is a, there is no, is in, is not in, is among, is not among, contains), data type operators (is a, is not a), geometry operators (is within, is not within), and basic logical operators (and, or, not).

Binary versus unary operators:

Operators can use either one argument (a unary operator) or two arguments (a binary operator):

"a" & "b" -- & is binary
there is a field "c" -- there is a is unary

The bitNot, there is a, there is no, is a, is not a, and not operators are unary operators. All other operators are binary operators.

Conversion of values:

Transcript converts values in expressions to whatever type of data is needed for the operation. This conversion happens automatically, so you don't need to do what in other languages is called *type casting*.

For example, suppose you have the following statement:

```
put char 2 of "123" + char 3 of "456" into field 3
```

Character 2 of the literal string `"123"` is the single-character string `"2"`, and character 3 of the literal string `"456"` is the single-character string `"6"`. When Transcript uses the `+` operator, it automatically converts these strings to numbers so that they can be added together. Then it converts the resulting number back to a string so that it can be placed in a field as text.

Operator precedence

When you combine sources of value using operators, Transcript evaluates each source of value in the expression. Next, it applies any operators to arrive at a final value for the expression. Transcript does not necessarily apply all the operators in right-to-left order.

Instead, it uses the operator precedence order to determine how to compute expressions that include more than one operator. Precedence determines the order in which Transcript carries out calculations in expressions. If an expression contains more than one operator, the operators with higher precedence are calculated before operators with lower precedence.

Tip: To see a list of all operators and their precedence, choose *Operator Precedence Reference* from the See Also menu above.

The grouping operator ():

Suppose you want to change the precedence used in an expression? The grouping operator `()` has higher precedence than all other operators, so you can use it to change the order in which operators are evaluated. In other words, if you enclose part of an expression in parentheses, any operations inside the parentheses are evaluated before anything else happens.

For example, function calls have higher precedence than division, so the expression `the sin of 1/4` means *get the sine of 1, then divide it by 4*:

```
get the sin of 1/4 -- does the sin function first
```

If you want the sine of $1/4$, you need to change the order of evaluation so that the division is done first. You do this by surrounding the part you want evaluated first with parentheses:

```
get the sin of (1/4) -- does the division first
```

If parentheses are nested, the expression within the innermost set of parentheses is evaluated first.

Factors and expressions

An expression is any source of value, or combination of sources of value. Any of the sources of value discussed above—containers, properties, function calls, literal strings, and constants—is a simple expression. You use operators to combine sources of value to produce more complex expressions.

Defining factors:

A factor is the first fully resolvable portion of an expression. (All factors are expressions, but not all expressions are factors.) A factor can be either a source of value, or an expression that combines sources of value but doesn't include any binary operators outside parentheses. Enclosing an expression in parentheses turns it into a factor.

These examples show some expressions, along with the first factor in each expression, to help you see the distinction:

Expression:	Its first factor:
3 + 4	3
(3 + 4)	(3 + 4)
(3 + 4)/field 4	(3 + 4)
field 6 * pi	field 6
sin of pi/4	sin of pi
sin(pi/4)	sin(pi/4)
sin(pi/4) * exp2(12)	sin(pi/4)
whole world	whole
"whole world"	"whole world"

When it matters:

The distinction between factors and expressions matters when you're using the `of` form of built-in functions, when you use URLs, and when you refer to objects.

• If you use the `of` form of a built-in function that has a parameter, the parameter must be a factor, not an expression:

```
get the sqrt of 4 + 5 -- yields 7
get sqrt(4+5) -- yields 3
get the sqrt of (4 + 5) -- yields 3
```

In the first example above, although we intended to get the square root of 9, we actually ended up with the square root of 4. This is because the expression `4 + 5` is not a factor (because it contains a binary operator that's not inside parentheses). The first factor in the expression `4 + 5` is `4`, so the first example gets the square root of 4 (which is 2), then adds 5 to that result.

The second example avoids this problem because it doesn't use the `of` form of the function. Since the parameter is enclosed in parentheses, you can use either a factor or an expression, and obtain the intended result, the square root of 9.

In the third example, we turn the expression $4 + 5$ into a factor by surrounding it with parentheses. Since the parameter is now a factor, we can get the correct result, even using the `the` form of the function.

ò When referring to URLs, the URL is a factor:

```
get URL "file:myfile.txt" -- works
get URL "file:" & "myfile.txt" -- DOESN'T WORK
get URL ("file:" & "myfile.txt") -- works
```

In the first example, the URL specified is a factor because it is a simple string with no operators.

The URL in the second example is not a factor, because it includes the binary operator `&`, so the `get` command tries to get the URL `file:` which is nonexistent and concatenate the content of that URL with the string `myfile.txt`.

In the third example, we turn the URL into a factor by surrounding it with parentheses, providing the expected result.

ò When referring to cards or backgrounds, the name, number or ID of the object is an expression:

```
go card 1 + 1 -- goes to card 2
go card 3 of background "Data" && "Cards"
-- goes to first card with the group "Data Cards"
```

However, when referring to controls (including groups) or stacks, the name, number, or ID of the object is a factor:

```
answer field 1 + 10 -- displays field 1 content + 10
answer field (1 + 10) -- displays field 11 content
select button "My" && "Button" -- DOESN'T WORK
select button ("My" && "Button") -- works
```

Summary

In this topic, you have learned that:

- ò A container holds data that you can change (with the `put` command) and use in an expression. Revolution has seven kinds of containers: variables, fields, buttons, images, URLs, the message box, and the selection.
- ò You can change the entire content of a container, or use chunk expressions to change a portion of it.
- ò A variable is a named container you create. To put something into a variable or check its contents, you refer to the variable by name.
- ò Fields, buttons, and images are all containers, as well as objects: you can put data into them and get their content.
- ò URLs are containers that refer either to a local file or to a file on another system that's accessible via the Internet. You use the URL keyword to use a URL as a container.

ò Using a URL on another system downloads the file, and putting something into a URL on another system uploads the data.

ò You can use properties, function calls, literal strings, and constants as sources of value in an expression. However, you can't put data into these sources of value, so they are not containers.

ò You use operators to combine sources of value into more complex expressions.

About using URLs, uploading, and downloading

See Also:

About chunk expressions, About filename specifications and file paths, How to access the Internet from behind a firewall, How to cancel a file transfer in progress, How to convert line endings when transferring a file, How to copy a resource fork, How to create a directory on an FTP server, How to create a file, How to display a picture from a web server, How to display a web page in a field, How to download a file from an FTP server, How to export data to a file, How to fetch data from the Internet, How to import data from a file, How to list the files in an FTP directory, How to open a URL in a browser, How to remove a file from an FTP server, How to upload a file to an FTP server, How to unwedge upload and download operations, How to use a stack thatÆs on a web server, Why canÆt I open a downloaded stack?, Why canÆt I upload a file?, Why donÆt URLs work in a standalone?, Why is there a problem with line endings?, Why was a downloaded file corrupted?, `binfile` keyword, `cachedURLs` function, `delete URL` command, `file` keyword, `ftp` keyword, `get` command, `go` command, `http` keyword, `libURLDownloadToFile` command, `libURLftpUpload` command, `libURLftpUploadFile` command, `load` command, `post` command, `put` command, `resfile` keyword, `URL` keyword, `URLStatus` function

A URL is a container for a file (or other resource), which may either on the same system the application is running on, or on another system thatÆs accessible via the Internet.

This topic discusses the various URL schemes that Transcript implements, how to create and manipulate files using URLs, and how to transfer data between your system and an FTP or HTTP server.

To fully understand this topic, you should know how to create objects and write short scripts, and understand how to use variables to hold data. You should also have a basic understanding of how the Internet works. If you have gone through the *Getting Started* tutorial, have read the topic *About containers, variables, and sources of value*, and have used a web browser, you have enough information to fully understand this topic.

Contents:

- An Overview of URLs

- URL Schemes

- Using URLs

- Creating and Deleting URLs

- Uploading and Downloading Files

- Summary

An Overview of URLs

In the Transcript language, a URL is a container for a file (or other document, such as the output of a CGI on a web server). The data in a URL may be on the same system the application is running on, or may be on another system, depending on the URLÆs scheme.

Like variables, URLs have no visual representation. However, unlike variables, URLs are persistent: if you make a change to a URL container, quit the application and then start it up again, then get the contents of the URL again, your change is included.

This is not surprising, since URLs, unlike all other containers, are external to your application. A URL is not a part of any of your stacks, but is stored separately.

URLs in Revolution are written like the URLs you see in a browser. You use the URL keyword to designate a URL, enclosing the URL's name in double quotes:

```
put field "Info" into URL "file:myfile.txt"
get URL "http://www.example.org/stuff/nonsense.html"
  put URL "ftp://ftp.example.net/myfile" into field "Data"
```

URL Schemes

A URL scheme is a type of URL. Revolution supports five URL schemes with the URL keyword: http, ftp, file, binfile, and (on Mac OS and OS X) resfile.

The http and ftp schemes designate documents or directories that are located on another system that's accessible via the Internet. The file, binfile, and resfile schemes designate local files.

The http scheme

An http URL designates a document from a web server:

```
put URL "http://www.example.org/home.html" into field "Page"
```

When you use an http URL in an expression, Revolution downloads the URL from the server and substitutes the downloaded data for the URL.

When you put something into an http URL, Revolution uploads the data to the web server:

```
put field "Info" into URL "http://www.example.net/info.html"
```

Note: Because most web servers do not allow HTTP uploads, putting something into an http URL usually will not be successful. (Check with the server's administrator to find out whether you can use the HTTP protocol to upload files.)

For more details about http URLs, see the entry for the http keyword in the Transcript Dictionary.

The ftp scheme

An ftp URL designates a file or directory on an FTP server:

```
get URL "ftp://user:passwd@ftp.example.net/picture.jpg"
```

When you use an ftp URL in an expression, Revolution downloads the URL from the server and substitutes the downloaded data for the URL.

When you put something into an ftp URL, Revolution uploads the data to the FTP server:

```
put image 10 into
```

```
URL "ftp://user:passwd@ftp.example.net/picture.jpg"
```

FTP servers require a user name and password, which you can specify in the URL. If you don't specify a user name and password, Revolution adds the "anonymous" user name and a dummy password automatically, in accordance with the convention for public FTP servers.

Note: Uploading to an FTP server usually requires a registered user name and password.

For more details about ftp URLs, see the entry for the ftp keyword in the Transcript Dictionary.

Directories on an FTP server:

A URL that ends with a slash (/) designates a directory (rather than a file). An ftp URL to a directory evaluates to a listing of the directory's contents.

The file scheme

A file URL designates a file on your system:

```
put field "Stuff" into URL "file:/Disk/Folder/testfile"
```

When you use a file URL in an expression, Revolution gets the contents of the file you designate and substitutes it for the URL. The following example puts the contents of a file into a variable:

```
put URL "file:myfile.txt" into myVariable
```

When you put data into a file URL, Revolution puts the data into the file:

```
put myVariable into URL "file:/Volumes/Backup/data"
```

Note: As with local variables, if the file doesn't exist, putting data into it creates the file.

File path syntax and the file scheme:

The file URL scheme uses the same file path syntax used elsewhere in Transcript statements. You can use both absolute paths and relative paths in a file URL.

For more information about file paths, see the topic "About filename specifications and file paths".

Conversion of end-of-line markers:

Different operating systems use different characters to mark the end of a line. Mac OS and OS X use a return character (ASCII 13), Unix systems use a linefeed character (ASCII 10), and Windows systems use a return followed by a linefeed. To avoid problems when transporting a stack between platforms, Revolution always uses linefeeds internally.

When you use a file URL as a container. Revolution translates as needed between the your system's end-of-line marker and Revolution's linefeed character.

The binfile scheme

A binfile URL designates a file on your system that contains binary data:

```
put URL "binfile:beachball.gif" into image "Beachball"
```

When you use a binfile URL in an expression, Revolution gets the contents of the file you designate and substitutes it for the URL. The following example puts the contents of a file into a variable:

```
put URL "binfile:picture.png" into pictVar
```

When you put data into a binfile URL, Revolution puts the data into the file:

```
put pictVar into URL "binfile:/Volumes/Backup/pict.png"
```

As with local variables, if the file doesn't exist, putting data into it creates the file.

File path syntax and the binfile scheme:

The binfile URL scheme uses the same file path syntax used elsewhere in Transcript statements. You can use both absolute paths and relative paths in a binfile URL.

For more information about file paths, see the topic [About filename specifications and file paths](#).

End-of-line conversion, the file scheme, and the binfile scheme:

The binfile scheme works like the file scheme, except that Revolution does not attempt to convert end-of-line markers.

This is because return and linefeed characters can be present in a binary file but not be intended to mark the end of the line. Changing these characters can corrupt a binary file, so the binfile scheme leaves them alone.

The resfile scheme

On Mac OS and OS X systems, files consist of either or both of a data fork and a resource fork. The resource fork contains defined resources such as icons, menu definitions, dialog boxes, fonts, and so forth. A resfile URL designates the resource fork of a Mac OS or OS X file:

```
put myBinaryData into URL "resfile:/Disk/Resources"
```

When you use a resfile URL in an expression, Revolution gets the resource fork of the file you designate and substitutes it for the URL.

When you put data into a resfile URL, Revolution puts the data into the file's resource fork.

Note: A resfile URL specifies the entire resource fork, not just one resource.

The most common use for this URL scheme is to copy an entire resource fork from one file to another. To modify the data from a resfile URL, you need to understand the details of Apple's resource fork format.

File path syntax and the resfile scheme:

The resfile URL scheme uses the same file path syntax used elsewhere in Transcript statements. You can use both absolute paths and relative paths in a resfile URL.

For more information about file paths, see the topic [About filename specifications and file paths](#).

Creating a resource fork:

Unlike the file and binfile URL schemes, the resfile keyword cannot be used to create a file. If the file doesn't yet exist, you cannot use the resfile keyword to create it. To create a new resource file, first use a file URL to create the file with an empty data fork, then write the needed data to its resource fork:

```
put empty into URL "file:myFile" -- creates an empty file
put myStoredResources into URL "resfile:myFile"
```

Using URLs

You use a URL like any other container. You can get the content of a URL or use its content in any expression. You can also put any data into a URL.

Note: URLs can hold either text or binary data:

- ò http, ftp, binfile, and resfile URLs can hold binary data.
- ò http, ftp, and file URLs can hold text.

The URL keyword

To specify a URL container, you use the URL keyword before the URL, which can use any of the five schemes described above:

```
if URL "http://www.example.net/index.html" is not empty...
get URL "binfile:/Applications/Hover.app/data"
put 1+1 into URL "file:output.txt"
```

The URL keyword tells Revolution that you are using the URL as a container.

Note: Some properties (such as the filename of a player or image) let you specify a URL as the property's value. Be careful not to include the URL keyword when specifying such properties, because using the URL keyword indicates that you're treating the URL as a container. If you use the URL keyword when specifying such a property, the property is set to the contents of the URL, not the URL itself, and this is usually not what's wanted.

Using the content of a URL

As with other containers, you use the content of a URL by using a reference to the URL in an expression. Transcript substitutes the URL's content for the reference.

If the URL scheme refers to a local file (file, binfile, or resfile URLs), Revolution reads the content of the file and substitutes it for the URL reference in the expression:

```
answer URL "file:../My File" -- displays the file's content
put URL "binfile:flowers.jpg" into myVariable
put URL "resfile:Icons" into URL "resfile:New Icons"
```

If the URL scheme refers to a document on another system (http or ftp URLs), Revolution downloads the URL automatically, substituting the downloaded data for the URL reference:

```
answer URL "http://www.example.net/files/greeting.txt"
```

Note: If the server sends back an error message—for example, if the file you specify in an http URL doesn't exist—then the error message replaces the URL reference in the expression.

Important! When you use an ftp or http URL in an expression, the handler pauses until Revolution is finished downloading the URL. However, other handlers can be executed during this time. Any other blocking URL operation (using the put command to upload a URL, the post command, the delete URL command, or a statement that gets an ftp or http URL) you try to execute before the URL download is completed will fail.

Putting data into a URL

As with other containers, you can put data into a URL. The result of doing so depends on whether the URL scheme specifies a file on your system (file, binfile, or resfile) or on another system (http or ftp).

If the URL scheme refers to a local file (file, binfile, or resfile URLs), Revolution puts the data into the specified file:

```
put field "My Text" into URL "file:storedtext.txt"
put image 1 into URL "binfile:picture.png"
```

If the URL scheme refers to a document on the Internet (http or ftp URLs), Revolution uploads the data to the URL:

```
put myVar into URL "ftp://me:pass@ftp.example.net/file.dat"
```

(Because most web servers do not allow HTTP uploads, this usually will not be successful with the http scheme.)

Important! When you put data into an ftp or http URL, the handler pauses until Revolution is finished uploading the URL. However, other handlers can be executed during this time. Any other blocking URL operation (using the put command to upload a URL, the post command, the delete URL command, or a statement that gets an ftp or http URL) you try to execute before the URL upload is completed will fail.

Chunk expressions and URLs

Like other containers, URLs can be used with chunk expressions to specify a portion of what's in a URL—a line, an item, a word, or a character. In this way, any chunk of a URL is like a container itself.

For more information about chunk expressions, see the topic "About chunk expressions".

Getting a chunk of a URL:

You can use any chunk of a URL in an expression, in the same way you use a whole URL:

```
get line 2 of URL "http://www.example.net/index.html"
put word 8 of URL "file:/Disk/Folder/myfile" into field 4
if char 1 of URL "ftp://ftp.example.org/test.jpg" is "0"...
```

You can also specify ranges, and even one chunk inside another:

```
put char 1 to 30 of URL "binfile:/marks.dat" into myVar
answer line 1 to 3 of URL "http://www.example.com/file"
```

Putting data into a chunk:

If the URL is local (that is, if it is a file, binfile, or resfile URL), you can put a value into a chunk of the URL:

```
put it into char 7 of URL "binfile:/picture.gif"
put return after word 25 of URL "file:../datafile"
put field 3 into line 20 of URL "file:myfile.txt"
```

You can also put a value into a chunk of an ftp or http URL. Because it's impossible to upload part of a file, Revolution downloads the file, makes the change, then uploads the file back to the server.

This is inefficient if you need to make several changes. In this case, it's faster to first put the URL in a variable, replace the chunk you want to change, then put the variable into the URL:

```
put URL "ftp://me:secret@ftp.example.net/file.txt" into myVar
put field "New Info" after line 7 of myVar
put field "More" into word 22 of line 3 of myVar
put myVar into URL "ftp://me:secret@ftp.example.net/file.txt"
```

This ensures that the file only needs to be downloaded once and re-uploaded once, no matter how many changes you need to make.

HTTP methods and http URLs

The basic operations used by the HTTP protocol are called methods. For http URLs, the following HTTP methods are used under the following circumstances:

- ò GET: when an http URL in an expression is evaluated
- ò PUT: when you put a value into an http URL
- ò POST: when you use the post command
- ò DELETE: when you use the delete URL command with an http URL

Note: Many HTTP servers do not implement the PUT and DELETE methods, which means that you can't put values into an http URL or delete an http URL on such servers. It's common to use the FTP protocol instead to upload and delete files; check with your server's administrator to find out what methods are supported.

HTTP headers:

When Revolution issues a GET or POST request, it constructs a minimal set of HTTP headers. For example, when issued on a Mac OS system, the statement:

```
put URL "http://www.example.org/myfile" into myVariable
```

results in sending a GET request to the server:

```
GET /myfile HTTP/1.1
Host: 127.0.0.0
User-Agent: Revolution (MacOS)
```


You can add headers, or replace the Host or User-Agent header, by setting the HTTPHeaders property before using the URL:

```
set the HTTPHeaders to "User-Agent: MyApp"

& return & "Connection: close"
put URL "http://www.example.org/myfile" into myVariable
```

Now the request sent to the server looks like this:

```
GET /myfile HTTP/1.1
Host: 127.0.0.0
User-Agent: MyApp
Connection: close
```

URLs and memory

URLs, unlike other containers, are only read into memory when you use the URL in a statement. (Other containers—like variables, fields, buttons, and images—are normally kept in memory, so using them doesn't increase memory usage.)

This means that in order to read a URL or place a value in a chunk of a URL, Revolution reads the entire file into memory. Because of this, you should be cautious when using a URL to refer to any very large file.

Even when referring to a single chunk in a URL, Revolution must place the entire URL in memory. An expression such as line 347882 of URL "file:bigfile.txt" may be evaluated very slowly or even not work at all, if not enough memory is available. If you refer to a chunk of an ftp or http URL, Revolution must download the entire file to find the chunk you specify.

Tip: To work with a file that's too large to fit into memory, use the open file, read from file, write to file, and close file commands to work with the data in smaller, more manageable pieces, instead of using URL references to the file.

The URL cache

As noted above, when you refer to an ftp or http URL, Revolution downloads the URL. Waiting for the download may take enough time to be annoying, particularly if the document is large, the user's connection is slow, or the context requires a quick response from the application.

The load command downloads the specified document and places it in a cache. Once a document has been cached, it can be accessed nearly instantaneously when you use its URL, because Revolution uses the cached copy in memory instead of downloading the URL again.

To use a file that has been downloaded by the load command, refer to it using the URL keyword as usual. When you request the original URL, Revolution uses the cached file automatically.

For best performance, use the load command at a time when response speed isn't critical (such as when your application is starting up), and only use it for documents that must be displayed quickly, such as images from the web that will be shown when you go to the next card.

The load command is non-blocking, so it does not stop the current handler while the download is completed. The handler continues while the load command downloads the URL in the background.

Creating and Deleting URLs

Creating a URL is similar to creating a local variable. You create a URL by putting something into it:

```
put myVariable into URL "file:newfile.txt"
```

If the file `newfile.txt` doesn't already exist, the statement above creates it.

Tip: On Mac OS and OS X systems, you specify a new file's creator signature and file type by setting the `fileType` property before creating the file.

URL schemes and URL creation:

Whether you can use a URL to create a file depends on the URL scheme.

The file and binfile URL schemes can be used to create a new file on your system:

```
put field "New Stuff" into URL "file:/Disk/Folder/New Stuff"
put image "Picture" into URL "../pix/flower.jpg"
```

The resfile URL scheme can create a resource fork in an existing file, but it cannot create a file. To create a new resource file, first use the file scheme to create the file, then use the resfile scheme to add the resources:

```
put empty into URL "file:myFile" -- creates an empty file
put myStoredResources into URL "resfile:myFile"
```

The ftp URL scheme can be used to create a new file to an FTP server. As with the file and binfile schemes, putting something into the URL creates the file:

```
put dataToUpload
into URL "ftp://jane:pass@ftp.example.com/newfile.dat"
```

Tip: You can create an FTP directory by uploading a file to the new (nonexistent) directory. The directory is automatically created. You can then delete the file, if you wish, leaving a new, empty directory on the server:

```
-- Create an empty file in the nonexistent directory:
put empty into
URL "ftp://jane:pass@example.com/newdir/dummy"
-- Delete unwanted empty file to leave new directory:
delete URL "ftp://jane:pass@example.com/newdir/dummy"
```

The http URL scheme can be used to create a new file on an HTTP server, by putting data into the URL. In practice, however, most servers are not configured to allow uploading files via HTTP; usually, you use the FTP protocol instead. Check with the server's administrator to find out how to upload files.

Deleting URLs:

You remove a URL with the delete URL command.

To delete a local file, you use a file or binfile URL:

```
delete URL "file:C:/My Programs/test.exe"  
delete URL "binfile:../mytext.txt"
```

It doesn't matter whether the file contains binary data or text; for deletion, these URL schemes are equivalent.

Tip: You can also use the delete file command to remove a file.

To delete the resource fork of a file, you use a resfile URL. The following example removes the resource fork along with all resources, but leaves the file in place:

```
delete URL "resfile:/Volumes/Backup/proj.rev"
```

Tip: To delete a single resource instead of the entire resource fork, use the deleteResource function.

To remove a file or directory from an FTP server, you use an ftp URL:

```
delete URL "ftp://root:secret@ftp.example.org/delete-me.txt"  
delete URL "ftp://me:mime@ftp.example.net/trash/"
```

As with creating files, you can use an http URL to delete a file, but most HTTP servers are not configured to allow this.

Uploading and Downloading Files

The simplest way to transfer data to an FTP or HTTP server is to use the put command to upload, or use the URL in an expression to download.

The Internet library includes additional commands to upload and download files to and from an FTP server. These commands offer more versatile options for monitoring and controlling the progress of the file transfer.

Uploading using the put command

As mentioned above, putting something into an ftp or http URL uploads the data to the server:

```
put myVariable  
  
into URL "ftp://user:pass@ftp.example.org/newfile.txt"
```

If you use the put command with a file or binfile URL as the source, the file is uploaded:

```
put URL "file:newfile.txt"  
  
into URL "ftp://user:pass@ftp.example.org/newfile.txt"
```

When you upload data in this way, the operation is blocking: that is, the handler pauses until the upload is finished. If there is an error, the error is placed in the result function:

```
put field "Data" into URL myFTPDestination
if the result is not empty then beep 2
```

Important! Uploading a URL does not prevent other messages from being sent during the file transfer: the current handler is blocked, but other handlers are not. For example, the user might click a button that uploads or downloads another URL while the first URL is still being uploaded. In this case, the second file transfer is not performed and the result is set to `ôError Previous request has not completed.ö`

To avoid this problem, you can set a flag while a URL is being uploaded, and check that flag when trying to upload or download URLs to make sure that there is not already a file transfer in progress.

Downloading using a URL

Referring to an ftp or http URL in an expression downloads the document.

```
put URL "ftp://ftp.example.net/myfile.jpg" into image 1
get URL "http://www.example.com/newstuff/newfile.html"
```

If you use the put command with a file or binfile URL as the destination, the document is downloaded to the file:

```
put URL "ftp://ftp.example.net/myfile.jpg"
    into URL "binfile:/Disk/Folder/myfile.jpg"
```

As with uploading using URLs, the download operation is blocking, and the handler pauses until the download is finished. If there is an error, the error is placed in the result function.

Other commands for uploading and downloading files

The Internet library provides a number of commands for transferring files via FTP:

`libURLftpUpload`: uploads data to an FTP server

`libURLftpUploadFile`: uploads a file to an FTP server

`libURLDownloadToFile`: downloads a file from an FTP server to a local file

The basic effect of these commands is the same as the effect of using URLs: that is, the data is transferred to or from the server. The following sets of statements each show one of the Internet library commands, with the equivalent use of a URL:

```
libURLftpUpload myVar,"ftp://me:pass@example.net/file.txt"
put myVar into URL "ftp://me:pass@example.net/file.txt"
```

```
libURLftpUploadFile "test.data","ftp://ftp.example.org/test"
put URL "binfile:test.data"
```

```
    into URL "ftp://ftp.example.org/test"
```

```
libURLDownloadToFile "ftp://example.org/new_beta","/HD/File"  
put URL "ftp://example.org/new_beta"  
  
into URL "binfile:/HD/File"
```

While the end result of using these library commands is the same as the end result of using URLs, there are several differences in how the actual file transfer is handled. Because of these differences, the library commands are often more suitable for uploads and downloads, particularly if the file being transferred is large.

Blocking and non-blocking file transfers:

When you transfer a file using URL containers, the file transfer stops the current handler until the transfer is done. This kind of operation is called a blocking operation, since it blocks the current handler as long as itÆs going on.

The Internet library commands, however, are non-blocking: the handler continues as soon as the command is issued. There is no pause in operation, although the file transfer takes the same amount of time. Non-blocking file transfers have several advantages:

- ò Since contacting a server may take some time due to network lag, the pause involved in a blocking operation may be long enough to be noticeable to the user.

- ò If a blocking operation involving a URL is going on, no other blocking operation can start until the previous one is finished. If a non-blocking file transfer is going on, however, you can start other non-blocking file transfers. This means that if you use the library commands, the user can begin multiple file transfers without errors.

- ò During a non-blocking file transfer, you can check and display the status of the transfer. This lets you display the transferÆs progress and allow the user to cancel the file transfer.

Checking status during a file transfer:

While a file is being transferred using the library commands, you can check the status of the transfer using the URLStatus function. This function returns the current status of a URL thatÆs being downloaded or uploaded:

```
put the URLStatus of "ftp://ftp.example.com/myfile.txt"  
  
into field "Current Status"
```

The URLStatus function returns one of the following values:

- òqueuedö: on hold until a previous request to the same site is completed
- òcontactedö: the site has been contacted but no data has been sent or received yet
- òrequestedö: the URL has been requested
- òloading bytesTotal,bytesReceivedö: the URL data is being received
- òuploading bytesTotal,bytesReceivedö: the file is being uploaded to the URL
- òcachedö: the URL is in the cache and the download is complete
- òuploadedö: the application has finished uploading the file to the URL

ôerrorö: an error occurred and the URL was not transferred
ôtimeoutö: the application timed out when attempting to transfer the URL

To monitor the progress of a file transfer, you check the URLStatus function at various times during the transfer.

Using callback messages:

When you start a file transfer using the libURLftpUpload, libURLftpUploadFile, or libURLDownloadToFile command, you can optionally specify a callback message, which is usually a custom message that you write a handler for. This message is sent whenever the file transferÆs URLStatus changes, so you can handle the callback message to handle errors or to display the file transferÆs status to the user.

The following simple example demonstrates how to display a status message to the user. The following handlers might be found in a buttonÆs script:

```
on mouseUp
  libURLDownloadToFile "ftp://example.org/new_beta",

    "/HD/Latest Beta","showStatus"
end mouseUp

on showStatus theURL
  put the URLStatus of theURL into field "Status"
end showStatus
```

When you click the button, the mouseUp handler is executed. The libURLDownloadToFile command begins the file transfer, and its last parameter specifies that a ôshowStatusö message will be sent to the button whenever the URLStatus changes.

As the URLStatus changes periodically throughout the download process, the buttonÆs ôshowStatusö handler is executed repeatedly. Each time a ôshowStatusö message is sent, the handler places the new status in a field. The user can check this field at any time during the file transfer to see whether the download has started, how much of the file has been transferred, and whether there has been an error.

Canceling a file transfer:

If a file transfer was started using the libURLftpUpload, libURLftpUploadFile, or libURLDownloadToFile command, you can cancel the transfer using the unload command:

```
unload URL "ftp://example.org/new_beta"
```

Uploading, downloading, and memory:

When you use a URL as a container, Revolution places the entire URL in memory. For example, if you download a file from an FTP server using the put command, Revolution downloads the whole contents of the file into memory before putting it into the destination container. If the file is too large to fit into available memory, a file transfer using this method will fail (and may cause other unexpected results).

The library commands libURLftpUpload, libURLftpUploadFile, and libURLDownloadToFile, however, do not require the entire file to be loaded into memory. Instead, they transfer the file one piece at a time.

If a file is (or might be) too large to comfortably fit into available memory, you should always use the library commands to transfer it.

Using a stack on a server

Ordinarily, you use stack files that are located on a local disk. You can also open and use a stack that is located on an FTP or HTTP server. Using this capability, you can update an application by downloading new stacks, make new functionality available via the Internet, and even keep most of your application on a server instead of storing it locally.

Going to a stack on a server:

As with local stack files, you use the go command to open a stack that's stored on a server:

```
go stack URL "http://www.example.org/myapp/main.rev"  
go stack URL "ftp://user:pass@example.net/secret.rev"
```

Note: For such a statement to work, the stack file must have been uploaded as binary data, uncompressed, and without encodings such as BinHex.

Revolution automatically downloads the stack file. The main stack of the stack file then opens in a window, just as though you had used the go command to open a local stack file.

You can go directly to a specific card in the stack:

```
go card "My Card"  
  
of stack URL "http://www.example.org/myapp/main.rev"
```

To open a substack instead, use the substack's name:

```
go stack "My Substack"  
  
of URL "http://www.example.org/myapp/main.rev"
```

Using a compressed stack:

You cannot directly open a stack that's compressed. However, since the stack URL is a container, you can use the URL as the parameter for the decompress function. The function takes the stack file data and uncompresses it, producing the data of the original stack file. You can open the output of the function directly as a stack.

The following statement opens a compressed stack file on a server:

```
go decompress(stack URL "http://www.example.net/comp.gz")
```

The statement automatically downloads the file `comp.gz`, uncompresses it, and opens the main stack of the file.

Saving stacks from a server:

When a stack is downloaded using the `go` command, it's loaded into memory, but not saved on a local disk. Such a stack behaves like a new (unsaved) stack until you use the `save` command to save it as a stack file.

Note: Saving a stack that has been downloaded with the `go` command does not re-upload it to its server. To upload a changed stack, you must save it to a local file, then use one of the methods described in this topic to upload the file to the server.

Summary

In this topic, you have learned that:

- ò A URL is a container, and you can use it like other containers: put data into it, get its contents, and examine or change a chunk of its data.

- ò Transcript supports five URL schemes: `file`, `binfile`, and `resfile` for local files, and `ftp` and `http` for documents on a server.

- ò As with local variables, you can create a URL by putting something into it. Putting data into a nonexistent `file` or `binfile` URL creates the file, and putting something into a nonexistent `ftp` or `http` URL uploads the data to a new file.

- ò You can transfer files to and from another system using URLs. Putting something into an `ftp` or `http` URL uploads the data to the document you specify. Getting an `ftp` or `http` URL downloads the specified document.

- ò You can also upload and download files using the commands in the Internet library. These commands, unlike the URL method, are non-blocking and offer more flexibility in monitoring the file transfer's status.

- ò To cache a document from a server for quick access, you use the `load` command.

- ò You can open a stack that's on a server by using the `stack file's` URL.

About chunk expressions

See Also:

About containers, variables, and sources of value, How to change every chunk in a container, Why is there a problem with line endings?, Recipe for reversing a string, character keyword, delete chunk command, item keyword, itemDelimiter property, line keyword, token keyword, word keyword

A chunk expression is a way of describing a specific portion of text in a container, such as a line or range of words.

This topic describes the syntax for specifying chunks and defines the types of chunks you can address.

To fully understand this topic, you should know how to write short scripts. If you have gone through the [Getting Started](#) tutorial, you have enough information to fully understand this topic.

Contents:

What is a Chunk?

Chunk Types

Specifying a Chunk

Specifying a Range

Checking the Existence of a Chunk

Summary

What is a Chunk?

A chunk is a precise description of the location of a piece of text in a container. You can use a chunk expressions to retrieve any part of a container—any specified character, word, or line, or any section delimited by a character you specify.

In fact, you can use a chunk of a container anywhere you use an entire container. For example, you can use the add command to add a number to a line of a field:

```
add 1 to word 3 of field "Numbers"
```

You can also use chunk expressions to replace (using the put command) or remove (using the delete chunk command) any portion of a container.

Using chunk expressions with properties:

You can use chunk expressions to read portions of a property (such as the script property). However, since you change a property with the set command rather than the put command, you can't use a chunk expression to change a part of a property's value.

Instead, put the property value into a variable, use the chunk expression to change the variable, then set the property to the variable's contents. The following example shows how to change the first line of an object's script property:

```
put the script of me into tempScript
put "-- Last changed by Jane" into line 3 of tempScript
```

set the script of me to tempScript

Chunk Types

A chunk expression specifies one or more chunks, which may be characters, tokens, words, items, or lines.

The character chunk

A character is a single character, which may be a letter, digit, punctuation mark, or control character.

A character cannot contain any other chunk type. It can be contained in any other chunk type.

You can use the abbreviation `char` as a synonym for `character` in a chunk expression.

Important! Characters in chunk expressions are assumed to be single-byte characters. To successfully use chunk expressions with Unicode (double-byte) text, you must treat each double-byte character as a set of two single-byte characters. For example, to get the numeric value of the third Unicode character in a field, use a statement like the following:

```
get charToNum(char 5 to 6 of field "Chinese Text")
-- char ((charNum* 2) - 1) to (charNum * 2) is
-- the single-byte equivalent of the double-byte
-- character "charNum" of the field.
```

The token chunk

A token is a string of characters delimited by certain punctuation marks. The token chunk is useful in parsing Transcript statements, and is generally used only for analyzing scripts. For full details about the definition of the token chunk, see the Transcript Dictionary.

A token can contain characters, but not any other chunk type. Tokens can be contained in a word, item, or line, but not in a character.

The word chunk

A word is a string of characters delimited by space, tab, or return characters or enclosed by double quotes.

A word can include characters, or tokens, but not items or lines. Words can be contained in a line or item, but not in a token or character.

The item chunk and the itemDelimiter property

By default, an item is a string of characters delimited by commas.

Items are delimited by the character in the `itemDelimiter` property. By default, the `itemDelimiter` is set to comma, but you can create your own chunk type by setting the `itemDelimiter` property to any character.

An item can contain characters, tokens, or words, but not lines. Items can be contained in a line, but not in a word, token, or character.

The line chunk and the lineDelimiter property

By default, a line is a string of characters delimited by the return character.

Lines are delimited by the character in the `lineDelimiter` property. By default, the `lineDelimiter` is set to `return`, but you can create your own chunk type by setting the `lineDelimiter` property to any character.

A line can contain characters, tokens, words, or items. Lines cannot be contained in any other chunk type.

Specifying a Chunk

The simplest chunk expression specifies a single chunk of any type. The following statements all include valid chunk expressions:

```
get char 2 of "ABC" -- yields "B"
get word 4 of "This is a test" -- yields "test"
get line 7 of myTestData
put "A" into char 2 of myVariable
```

You can also use the ordinal numbers `first`, `last`, `middle`, `second`, `third`, `fourth`, `fifth`, `sixth`, `seventh`, `eighth`, `ninth`, and `tenth` to designate single chunks. The special ordinal `any` specifies a random chunk.

```
put "7" into last char of "1085" -- yields "1087"
```

Negative indexes in chunk expressions

To count backwards from the end of the value instead of forward from the beginning, specify a negative number. For example, the number `-1` specifies the last chunk of the specified type, `-2` specifies the next-to-last chunk, and so forth. The following statements all include valid chunk expressions:

```
get item -1 of "feather, ball, cap" -- yields "cap"
get char -3 of "ABCD" -- yields "B"
```

Complex chunk expressions

More complex chunk expressions can be constructed by specifying a chunk within another chunk. For example, the chunk expression `word 4 of line 250` specifies the fourth word of line 250.

When combining chunks of different types to construct a complex chunk expression, you must specify the chunk types in order. The following statements all include valid chunk expressions:

```
char 7 of word 3 of myValue
word 9 of item 2 of myValue
last char of word 8 of line 4 of myValue
```

These, however, are not valid chunk expressions:

```
word 8 of char 2 of myValue -- chars can't contain words
item 9 of first word of myValue -- words can't contain items
line 3 of last item of myValue -- items can't contain lines
```

Using parentheses in chunk expressions

You use parentheses in chunk expressions for the same reasons they're used in arithmetic:

ò To make a complex expression clearer.

ò To change the order in which the parts of the expression are evaluated.

For example, consider the following statement:

```
put item 2 of word 3 of "a,b,c i,j,k x,y,z" -- BAD
```

The desired result is `ôyö`, the second item in the third word. But the statement above causes an execution error, because it asks for an item of a word, and words canÆt contain items. You can obtain the desired result by using parentheses to change the order of evaluation:

```
put item 2 of (word 3 of "a,b,c i,j,k x,y,z") -- good
```

In the example above, Transcript gets the third word first, then gets the second item in that word. By adding parentheses around `ôword 3 of "a,b,c i,j,k x,y,z"ö`, you force Transcript to evaluate that part of the chunk expression first. The value of the expression in parentheses is `ôx,y,zö`, and item 2 of `ôx,y,zö` is `ôyö`.

As with arithmetic expressions, the parts of a chunk expression that are in parentheses are evaluated first. If parentheses are nested, the chunk expression within the innermost set of parentheses is evaluated first. The part that is enclosed in parentheses must be a valid chunk expression, as well as being part of a larger chunk expression:

```
put line 2 of word 1 to 15 of myValue -- won't work
put line 2 of word (1 to 15 of myValue) -- won't work
put line 2 of word 1 to 15 (of myValue) -- won't work
put line 2 of (word 1 to 15 of myValue) -- works!
```

The first of the above examples doesnÆt work for much the same reason as the previous example: words canÆt contain lines. The second and third examples donÆt work because neither `ô1 to 15 of myValueö` nor `ôof myValueö` is a valid chunk expression. However, `ôword 1 to 15 of myValueö` is a valid chunk expression, so the last example works.

Nonexistent chunks

If you request a chunk number that doesnÆt exist, the chunk expression evaluates to empty. For example, the expression `char 7 of "AB"` yields empty.

If you attempt to change a chunk that doesnÆt exist, what happens depends on what kind of chunk you specify:

ò Nonexistent character or word:

Putting text into a character or word that doesnÆt exist appends the text to the end of the container, without inserting any extra spaces.

ò Nonexistent item:

Putting text into an item that doesnÆt exist adds enough itemDelimiter characters to bring the specified item into existence.

ò Nonexistent line:

Putting text into a line that doesn't exist adds enough return characters to bring the specified line number into existence.

Specifying a Range

To specify a portion larger than a single chunk, you specify the beginning and end of the range. These are all valid chunk expressions:

```
get char 1 to 3 of "ABCD" -- yields "ABC"
get word 2 to -1 of myValue -- second word to last word
put it into line 7 to 21 of myValue -- replaces
```

The start and end of the range must be specified as the same chunk type, and the beginning of the range must occur earlier in the value than the end. The following are not valid chunk expressions:

```
char 3 to 1 of myValue -- end cannot be greater than start
char -1 to -4 of myValue -- 4th from last comes before last
```

Important! To use negative numbers in a range, remember that numerically, -x comes after -x+1. For example, -1 is greater than -2, and 4 is greater than -7. The greater number must come last in order to create a valid range.

Checking the Existence of a Chunk

The number function returns the number of chunks of a given type in a value. For example, to find out how many lines are in a variable, use an expression such as:

```
the number of lines in myVariable
```

You can also nest chunk expressions to find the number of chunks in a single chunk of a larger chunk type:

```
the number of chars of item 10 of myVariable
```

The is among operator tells you whether a chunk exists in a larger value. For example, to find out whether the word ôfreeö is part of a larger string, use the is among operator:

```
"free" is among the words of "Live free or die" -- true
"free" is among the words of "Unfree world" -- false
```

The second example evaluates to false because, although the string ôfreeö is found in the value, it's a portion of a larger word, not an entire word.

Finally, the offset function, which locates the character position of a string within a larger string, can be used to determine the existence and location of a single character. For example, this expression returns the character number where the letter ôCö was found:

```
get offset("C","ABC") -- returns 3
```

The `lineOffset`, `itemOffset`, and `wordOffset` functions can be used similarly to locate lines, items, and words within a larger value.

Summary

In this topic, you have learned that:

- ò A chunk expression describes the location of a piece of text in a longer string.
- ò Chunk expressions can describe characters, items, tokens, words, and lines of text.
- ò To count backward from the end of a string, use negative numbers. For example, `word -2` indicates the second-to-last word.
- ò You can combine chunk expressions to specify one chunk that is contained in another chunk, as in `word 2 of line 3 of myVariable`.
- ò For a range of chunks, specify the start and end points of the range, as in `line 2 to 5 of myVariable`.
- ò To get a chunk expression specifying where certain text can be found in a container, use the `offset`, `lineOffset`, `itemOffset`, and `wordOffset` functions.

About properties and property profiles

See Also:

About containers, variables, and sources of value, About custom properties and custom property sets, Property Profiles Tutorial, How to change an object's property profile, How to change the profile for all the objects in a stack or card, How to display an object's property inspector, How to see and change an object's properties, How to store styled text in a variable or property, Object menu > Object Inspector, Object menu > Card Inspector, Object menu > Stack Inspector, properties property, propertyNames function, revProfile property, revSetCardProfile command, revSetStackFileProfile command, revSetStackProfile command

A property is an attribute of a Revolution object. Each type of object has many built-in properties, which affect the object's appearance or behavior. You can also define custom properties for any object, and use them to store any kind of data.

This topic discusses how to use properties, how properties are inherited between objects, and how to create and switch between collections of property settings.

To fully understand this topic, you should know how to create objects, how to use an object's property inspector, and how to write short scripts. If you have gone through the "Getting Started" tutorial, you have enough information to fully understand this topic.

Contents:

- Using Object Properties
- Property Inheritance
- Global Properties
- Chunk Properties
- Property Profiles
- Summary

Using Object Properties

A property is an attribute of an object, and each object type has its own set of built-in properties appropriate for that type. An object can be completely described by its built-in properties; if you could make all the properties of two objects identical, they'd be the same object.

Note: Since no two objects can have the same ID property, it's not possible in practice for two different objects to become the same object, because the ID will always be different.

Built-in properties determine the appearance and behavior of stacks and their contents—fonts, colors, window types, size and placement, and much more—as well as much of the behavior of the Revolution application. By changing properties, you can change almost any aspect of your application.

Referring to properties

Property references consist of the word "the", the property name, the word "of", and a reference to the object:

the armedIcon of button "My Button"

the borderWidth of field ID 2394
the name of card 1 of stack "My Stack"

Properties are sources of value, so you can get the value of a property by using it in an expression:

put the height of field "Text" into myVar
put the width of image "My Image" + 17 after field "Values"
if item 1 of the location of me > zero then beep

For example, to use the width property of a button as part of an arithmetic expression, use a statement like the following:

add the width of button "Cancel" to totalWidths

The value of the property—in this case, the width of the button in pixels—is substituted for the property reference when the statement is executed.

Tip: To see a list of all the properties for a particular object type, open the Documentation window, click Transcript Dictionary, and choose the object type from the menu at the top of the window.

Changing properties

To change the value of a property, you use the set command:

set the borderColor of group "My Group" to "red"
set the top of image ID 3461 to zero

You can also see and change many of an object's properties by selecting the object and choosing Object menu Object Inspector.

Most built-in properties affect the appearance or behavior of the object. For example, a button's height, width, and location are properties of the button. Changing these properties in a handler causes the button's appearance to change. Conversely, dragging or resizing the button changes the related properties.

Read-only properties:

Some properties can be read, but not set. These are called read-only properties. Trying to set a read-only property causes an execution error.

To find out whether a property is read-only, check its entry in the Transcript Dictionary.

Changing a part of a property:

Properties are not containers, so you cannot use a chunk expression to change a part of the property. (However, you can use a chunk expression to examine part of a property.) For example, you cannot set line 1 of a property to a new value: you must set the whole thing.

To change one part of a property, first put the property value into a variable, change the required part of the variable, then set the property back to the new variable contents:

put the rect of me into tempRect


```
put "10" into item 2 of tempRect
set the rect of me to tempRect
```

Custom properties

A custom property is a property that you define. You can create as many custom properties for an object as you want, and put any kind of data into them (even binary data). You can even store a file in a custom property.

For more information about custom properties, see the topic [About custom properties and custom property sets](#).

Property Inheritance

Most properties are specific to the object they are part of, and affect only that object.

However, some properties of an object, such as its color and text font, take on the settings of the object above it in the object hierarchy. For example, if a field's background color property is not specified (that is, if its backgroundColor property is empty), the field takes on the background color of the card that owns it. If no background color is specified for the card either, the stack's background color is used, and so on. This means you can set a background color for a stack, and every object in it will automatically use that background color, without your having to set it for each object.

This process of checking first the object, then the object's owner, then the object that owns that object, and so on, is called inheritance of properties. Each object inherits the background color of the object above it in the hierarchy. Similar inheritance rules apply to the foregroundColor, topColor, bottomColor, borderColor, shadowColor, and focusColor properties, to their corresponding pattern properties, and to the textFont, textSize, and textStyle properties.

Overriding inheritance

Inheritance is used to determine an object's appearance only if the object itself has no setting for the property. If an inheritable property of an object is non-empty, that setting overrides any setting the object might inherit from an object above it in the object hierarchy.

For example, if a button's backgroundColor property is set to a color reference instead of being empty, the button uses that background color, regardless of the button's owners. If the object has a color of its own, that color is always used.

The effective keyword

If an inheritable property of an object is empty, you can't simply check the property to find out what color or font settings the object displays. In this case, use the effective keyword to obtain the inherited setting of the property. The effective keyword searches the object's owners, if necessary, to find out what setting is actually used.

For example, suppose you have a field whose textFont property is empty. The textFont of the card that the field is on is set to `Helvetica`, so the field inherits this setting and displays its text in the Helvetica font. To find out what font the field is using, use the expression `the effective textFont`:

```
get the textFont of field "My Field" -- empty
get the effective textFont of field "My Field" -- Helvetica
```

You can use the effective keyword with any inherited property.

Global Properties

Revolution also has global properties, which affect the overall behavior of the application. Global properties are accessed and changed the same way as object properties. They do not belong to any particular object, but otherwise they behave like object properties.

Tip: To see a list of all global properties, open the Documentation window, click Transcript Dictionary, and choose "Global Properties" from the menu at the top of the window.

A few properties are both global and object properties. (For example, the paintCompression is a global property, and also a property of images.) For these properties, the global setting is separate from the setting for an individual object.

Referring to global properties

You refer to global properties using the and the property name:

```
the defaultFolder
the emacsKeyBindings
the fileType
```

Since global properties apply to the whole application, you don't include an object reference when referring to them.

Global properties are sources of value, so you can get the value of a global property by using it in an expression:

```
get the stacksInUse
put the recentNames into field "Recent Cards"
if the ftpProxy is empty then exit setMyProxy
```

Changing global properties

To change a global property, you use the set command, in the same way as for object properties:

```
set the itemDelimiter to "/"
set the grid to false
set the idleTicks to 10
```

Some global properties can be changed by other commands. For example, the lockScreen property can either be set directly, or changed using the lock screen and unlock screen commands. The following two statements are equivalent:

```
set the lockScreen to false -- does the same thing as...
unlock screen
```

Some other global properties are affected by system settings. For example, the default value of the playLoudness property is set by the operating system's sound volume setting.

Saving and restoring global properties

Object properties are part of an object, so they are saved when the stack containing their object is saved. Global properties, however, are not associated with any object, so they are not saved with a stack. If you change the value of a global property, the change is lost when you quit the application.

If you want to use the same setting of a global property during a different session of your application, you must save the setting in a Preferences file, in a custom property, or elsewhere in a saved file and restore it when your application starts up.

Chunk Properties

Normally, properties are applied only to objects or, in the case of global properties, to the entire application. However, a few properties also apply to chunks in a field or to single characters in a field.

Text style properties

Certain text-related properties can be applied either to an entire field or to a chunk of a field:

```
set the textFont of word 3 of field "My Field" to "Courier"
set the foregroundColor of line 1 of field 2 to "green"
if the textStyle of the clickChunk is "bold" then beep
```

The following field properties can be applied to either an entire field or to a chunk of the field:

```
ò textFont, textStyle, and textSize
ò textShift
ò backgroundColor and foregroundColor
ò backgroundPattern and foregroundPattern (Unix systems)
```

Each chunk of a field inherits these properties from the field, in the same way that fields inherit from their owners. For example, if a word's textFont property is empty, the word is displayed in the field's font. But if you set the word's textFont to another font name, that word and only that word is displayed in its own font.

To find the text style of a chunk in a field, whether that chunk uses its own styles or inherits them from the field, use the effective keyword:

```
get the effective textFont of word 3 of field ID 2355
answer the effective backgroundColor of char 2 to 7
of field "My Field"
```

Note: If a chunk expression includes more than one style, the corresponding property for that chunk reports `mixed`. For example, if the first line of a field has a textSize of `12`, and the second line has a textSize of `24`, an expression like the `textSize of line 1 to 2 of field "My Field"` reports `mixed`.

Formatted text properties

The `htmlText`, `RTFText`, and `unicodeText` properties of a chunk are equal to the text of that chunk, along with the formatting information that's appropriate for the property.

For example, if a field contains the text `This is a test.`, and the word `is` is boldfaced, the `htmlText` of word 2 reports `is`.

The formattedRect and related properties

The formattedRect property (along with the formattedWidth, formattedHeight, formattedLeft, and formattedTop) reports the position of a chunk of text in a field. These properties are read-only.

The formattedRect, formattedHeight, formattedLeft, and formattedTop properties can be used for a chunk of a field, but not the entire field. The formattedWidth and formattedHeight apply to both fields and chunks of text in a field.

The imageSource, linkText, and visited properties

The imageSource of a character specifies an image to be substituted for that character when the field is displayed. You use the imageSource to display images inside fields:

```
set the imageSource of char 17 of field 1 to 49232
set the imageSource of char thisChar of field "My Field"

to "http://www.example.com/banner.jpg"
```

The linkText property of a chunk lets you associate hidden text with part of a field's text. You can use the linkText in a linkClicked handler to specify the destination of a hyperlink, or for any other purpose.

The visited property specifies whether you have clicked on a text group during the current session. (You can get the visited property for any chunk in a field, but it is meaningless unless the chunk's textStyle includes `link`.)

The imageSource, linkText, and visited properties are the only properties that can be applied to a chunk of a field, but not to the entire field or any other object. (Because they are applied to text in fields, they are listed as field properties in the Transcript Dictionary.)

Property Profiles

A property profile is a collection of object property settings, which is stored as a set. A profile for an object can contain settings for almost any properties of the object.

You can include values for most built-in properties in a profile, and create as many different property profiles as you need for any object. Once you've created a profile, you can switch the object to the profile to change all the property values that are defined in the profile.

For example, suppose you create a property profile for a field that includes settings for the field's color properties. When you switch to that profile, the field's colors change, while all other properties (not included in the profile) remain the same.

Use property profiles when you want to:

- Create `skins` for your application
- Display your application in different languages
- Present different levels of `novice`, `expert`, and so on
- Use different user-interface standards for different platforms

Creating a property profile

You use the Property Profiles pane in an object's property inspector to create a profile for the object. Once you have created a profile, you can switch to it using either the property inspector or the commands in the Profile library.

To create a profile, follow these steps:

1. Open the object's property inspector and choose "Property Profiles" from the menu at the top of the inspector window.
2. In the "Profiles" section at the top, click the "New Profile" () icon.
3. Enter a profile name in the dialog box.

Profile names:

Profile names follow the same rules as variable names. A profile name must be a single word, consisting of letters, digits, and underscores, and must start with either a letter or an underscore.

Tip: If you want to use a single command to switch several objects to a particular profile, give the profile the same name for each of the objects it applies to.

The master profile:

Every object has a master profile that holds the default settings for all its properties. If you don't set an object's profile, the master profile is used. When you create a new profile, you can change settings for various properties to make them different from the master profile's settings.

If you don't specify a property setting in a profile, the master profile's setting is used, so you don't have to specify all properties of an object when you create a profile—only the ones you want to change.

By default, the master profile is named "Master". You can change the master profile's name in the "Property Profiles" pane of the Preferences window.

Creating a profile in a handler:

In addition to creating property profiles in the property inspector, you can create a profile in a handler.

To enable creating profiles, check the "Create profiles automatically" box in the "Property Profiles" pane of the Preferences window. If this box is checked, setting the `revProfile` property of an object automatically creates the profile.

This ability is particularly useful if you want to create a number of profiles, as shown in the following example:

```
on mouseUp
  -- creates a profile for each card in the stack
  repeat with thisCard = 1 to the number of cards
    set the revProfile of card x to "myNewProfile"
  end repeat
end mouseUp
```

The handler above creates a profile called `omyNewProfile` for all the cards in the current stack. (In order for this handler to work, the `Create profiles automatically` option in the `Property Profiles` pane of the Preferences window must be turned on.)

Updating property values in a profile

When you create a new profile, Revolution automatically switches the object to use that profile. There are two ways to include new property settings in a profile: by using the property inspector to specify the properties you want to include, and by changing the properties directly while the profile is active.

Adding profile settings in the property inspector:

In the Property Profiles pane, you can specify properties to include in a profile. To add a property setting to a profile, follow these steps:

1. If the profile is not already selected in the top section of the Property Profiles pane, click it to make it the active profile.
2. In the middle section of the Property Profiles pane, click the `New Property` () icon. The Add Property dialog box appears.
3. Choose the property you want to change from the list.

Tip: To show only properties that include a string, enter the string in the box at the top. For example, if you type `line`, the `lineSize` and `menuLines` properties appear.

4. The property you chose is added to the middle section, and selected. Enter the value that the property should have in the bottom section of the Property Profiles pane.

Important! The Add Property dialog box lists all applicable properties, but the Property Profiles pane automatically eliminates redundant properties and property synonyms. For example, if you add the `backgroundColor` and `foregroundColor` properties, the Property Profiles pane displays the `colors` property instead the next time you re-open the property inspector. This is because the `colors` property contains all eight color settings of an object, so it's not necessary to store the individual color properties once they've been set.

Adding profile settings in a handler:

You can add a property setting to a profile by switching to the profile, then setting the property:

```
set the revProfile of button 1 to "MyProfile"  
set the foregroundColor of button 1 to "red"  
set the revProfile of button 1 to "Master"
```

By default, if you change a property and then switch profiles, the property you changed and its current setting is saved with the profile.

Tip: You can control this behavior either in Preferences window or using the `gRevProfileReadOnly` keyword. If you don't want to save property changes when switching profiles, do one of the following:

- Set the `gRevProfileReadOnly` variable to true:

global gRevProfileReadOnly
put true into gRevProfileReadOnly

ò In the ôProperty Profilesö pane of the Preferences window, uncheck the box labeled ôDonÆt save changes in profileö.

The two methods of changing this setting are equivalent: changing the gRevProfileReadOnly variable also changes the preference setting, and vice versa.

Updating existing properties in a profile:

To change the setting of a property thatÆs already stored in a profile, follow these steps:

1. Open the Property Profiles pane of the objectÆs property inspector.
2. In the middle section of the Property Profiles pane, click the property you want to change.
3. In the bottom section of the Property Profiles pane, enter the propertyÆs new value.

Tip: To easily copy a property value from another profile, click the ôCopyö button in the bottom section and choose the profile you want to copy from.

As with adding new properties to a profile, you can update an existing property in a handler by switching to the profile, then setting the property to the new value. This behavior is also controlled by the gRevProfileReadOnly setting and the ôDonÆt save changes in profileö box in the ôProperty Profilesö pane of the Preferences window.

Switching between profiles

To switch between profiles, you can use either the property inspector or a handler. If several (or all) objects in a card, stack, or stack file have profiles with the same name, you can switch all the objects to that profile with a single command.

Switching a single object:

To switch an objectÆs profile, you can use either the objectÆs property inspector or the revProfile property.

In the property inspector, click the arrow icon at the upper right of the palette. The menu that appears includes a submenu called ôProperty Profilesö. To switch to a different profile, choose the profile you want from the submenu.

The arrow icon is available whenever the property inspector is visible, so you donÆt have to be using the Property Profiles pane to switch profiles. If the Property Profiles pane is visible, you can click the name of the profile to switch to it.

To switch an objectÆs profile in a handler or the message box, use a statement like the following:

```
set the revProfile of player "My Player" to "MyProfile"
```

Switching all the objects on a card:

To switch the profiles of all the objects on a card, use the revSetCardProfile command:

```
revSetCardProfile "MyProfile","My Stack"
```

The statement above sets the profile of all objects on the current card of the stack named `My Stack`. (Although the `revSetCardProfile` command changes a card, you specify a stack name, not a card name.)

If an object on the card does not have a profile with the specified name, the object is left untouched.

Switching all the objects in a stack:

To switch the profiles of all the objects in a stack, use the `revSetStackProfile` command:

```
revSetStackProfile "MyProfile","My Stack"
```

The statement above sets the profile of all objects in the stack named `My Stack`.

If an object in the stack does not have a profile with the specified name, the object is left untouched.

Switching all the objects in a stack file:

To switch the profiles of all the objects in every stack in a stack file, use the `revSetStackFileProfile` command:

```
revSetStackFileProfile "MyProfile","My Stack"
```

The statement above sets the profile of all objects in the stack named `My Stack`, along with any other stacks in the same stack file.

If an object in any of the stacks does not have a profile with the specified name, the object is left untouched.

Switching profiles in the Property Profiles pane:

You can also use the property inspector to switch all the profiles in a card, stack, or stack file. Follow these steps:

1. Select an object that has the profile you want to switch to, and open the object's property inspector.
2. Choose `Property Profiles` from the menu at the top of the inspector palette.
3. Select the profile you want to switch to, in the top section of the Property Profiles pane.
4. Click the `Set All` button at the top of the Property Profiles pane, then click `Card`, `Stack`, or `Stack File`.

Building standalone applications with property profiles

Because the commands that let you switch profiles are part of the Profile library, you must include the library (as well as the profiles you want to use) in your standalone application if you want to be able to switch profiles when the application runs outside the development environment.

You can also have the Distribution Builder automatically switch all the objects in your stack to a profile you choose when the application is built, whether or not you include the Profile library in your application.

Which of the options to use depends on how your application uses profiles. For example, if you use profiles to provide different appearance "skins", and the user can select a skin, you will need to include all these profiles plus the Profile library, because your application must switch profiles in order to change skins. But if you use profiles to adjust the application's appearance for each platform, you only need to include each platform's profile in the application for that platform.

Including the Profile library:

If your application uses the `revSetCardProfile`, `revSetStackProfile`, or `revSetStackFileProfile` command, or sets the `revProfile` property, you must include the Profile library when you create the application.

To include the Profile library, follow these steps:

1. Complete Step 1 and Step 2 in the Distribution Builder.
2. In Step 3, click the Profiles tab.
3. Make sure the "Include profiles and allow switching" option is selected.

Including property profiles:

If the "Include profiles and allow switching" option is selected, you can also choose which profiles to include in the application. If your application allows switching to a profile, you should include that profile. You include profiles in the Profiles tab in Step 3 of the Distribution Builder.

To include all the profiles for all objects, select the option labeled "Include all profiles on objects".

To include some profiles but not others, select the option labeled "Only include profiles selected below", then click the profiles you want to include. (Profiles you don't choose will be removed from the application when it's built, so you won't be able to switch to them.)

Note: The list includes every profile in the stack files you specified in Step 2 of the Distribution Builder, even if not all objects have that profile.

Applying a profile when you build an application:

Whether or not you include any profiles in your application, the Distribution Builder can automatically set all objects to the property settings contained in a profile you specify. For example, if you have a profile for each platform, you can automatically set all the objects to the correct profile when you build for that platform.

You apply a profile in the Profiles tab in Step 3 of the Distribution Builder:

1. On the Profiles tab, check the box labeled "Set all objects to profile".
2. From the menu below the checkbox, choose the profile you want.

When you click "Build Distribution", Revolution applies the property settings in the selected profile to all objects that have that profile. When you launch the application, the properties of all the objects that had the selected profile are pre-set to the profile's settings.

Important! You don't need to include the profile in your application in order to apply its settings during the build process. The Distribution Builder applies all the settings before it creates the new application.

Summary

In this topic, you have learned that:

- ò A property is an attribute of an object. Each object type has its own collection of properties, and taken together, the settings for all an object's properties describe everything about the object.
- ò You can use a property as a source of value in any expression.
- ò You use the set command to change the value of a property.
- ò Custom properties are properties that you create. A custom property can hold any data you want to put in it.
- ò Global properties apply to the entire application, not to a single object.
- ò You can create a property profile for an object to hold a collection of property settings. When you switch the object to a profile, all the profile's settings take effect. You can switch the profile of a single object, or all the objects in a card, stack, or stack file.

About custom properties and custom property sets

See Also:

About containers, variables, and sources of value, About properties and property profiles, Memory and Limits Reference, How to create a custom property, How to create a custom property set, How to delete a custom property, How to delete a custom property set, How to duplicate a custom property set, How to list an object's custom properties, How to refer to a custom property in a non-active set, How to rename a custom property, How to rename a custom property set, How to store an array in a custom property, How to store styled text in a variable or property, Why doesn't a custom property appear in the property inspector?, Object menu > Object Inspector, Object menu > Card Inspector, Object menu > Stack Inspector, customKeys property, customProperties property, customPropertySet property, customPropertySets property

A custom property is a property that you create for an object, in addition to its built-in properties. You can define custom properties for any object, and use them to store any kind of data.

This topic discusses how to create and use custom properties, how to organize custom properties into sets, and how to use `getProp` and `setProp` handlers to handle custom property requests.

To fully understand this topic, you should know how to create objects, how to use an object's property inspector, and how to write short scripts. You should also have a basic knowledge of how built-in properties work and know how arrays work. If you have gone through the "Getting Started" tutorial and have read the topic "About properties and property profiles", you have enough information to fully understand this topic.

Contents:

Using Custom Properties

Organizing Custom Properties

Attaching Handlers to Custom Properties

Virtual Properties

Summary

Using Custom Properties

A custom property is a property that you define. You can create as many custom properties for an object as you want, and put any kind of data into them (even binary data). You can even store a file in a custom property.

Use a custom property when you want to:

- ò associate data with a specific object
- ò save the data with the object in the stack file
- ò access the data quickly

Creating a custom property

You create a custom property by setting the new property to a value. If you set a custom property that doesn't exist, Revolution automatically creates the custom property and sets it to the requested value.

This means that you can create a custom property in a handler or the message box, simply by using the set command. The following statement creates a custom property called `endingTime` for a button:

```
set the endingTime of button "Session" to the long time
```

You can also view, create, and delete custom properties in the Custom pane of the object's property inspector. Follow these steps:

1. In the property inspector, choose `Custom Properties` from the menu at the top.
2. Click the New () button in the top section of the Custom Properties pane.
3. Enter a name for the new custom property.

You can create as many custom properties as you want.

Objects only:

You can create custom properties for any object.

However, you cannot create global custom properties, or custom properties for a chunk of text in a field. Unlike some built-in properties, a custom property applies only to an object.

Important! Each object can have its own custom properties, and custom properties are not shared between objects. Creating a custom property for one object does not create it for other objects.

Custom property names:

The name of a custom property must consist of a single word and may contain any combination of letters, digits, and underscores (`_`). The first character must be either a letter or an underscore.

Avoid giving a custom property the same name as a variable. If you refer to a custom property in a handler, and there is a variable by the same name, Revolution uses the contents of the variable as the name of the custom property. This usually causes unexpected results.

Important! Custom property names beginning with `rev` are reserved for Revolution's own custom properties. Naming a custom property with a reserved name may produce unexpected results when working in the development environment.

Referring to custom properties

Custom property references look just like built-in property references: the word `the`, the property name, the word `of`, and a reference to the object.

For example, to use a custom property called `lastCall` that belongs to a card, use a statement like the following:

```
put the lastCall of this card into field "Date"
```

Like built-in properties, custom properties are sources of value, so you can get the value of a custom property by using it in an expression. The property's value is substituted for the property reference

when the statement is executed. For example, if the card's `astCall` custom property is `Today`, the example statement above puts the string `Today` into the `Date` field.

The `customProperties` property:

The `customProperties` of an object consists of all the object's custom properties, in array form: each element of the array is a single custom property.

You can set (or get) all the custom properties of an object at once using the `customProperties` property. The following example copies the custom properties of one button to another button:

```
set the customProperties of button 2  
to the customProperties of button 1
```

Because the `customProperties` is an array, you can put it into a variable. To selectively change custom properties, it is sometimes convenient to move custom properties to a variable, change the elements of that variable, and then copy the variable to the `customProperties` of another object (or the original object). The following example gets the custom properties of a button, changes one custom property (called `MyProp`), then copies all the custom properties to a different button:

```
put the customProperties of button 1 into theCustomProps  
-- ...creates an array variable called "theCustomProps"  
-- Now we change one element of that variable:  
put "123" into theCustomProps["MyProp"]  
-- ...and set another button's properties to the copy:  
set the customProperties of button 2 to theCustomProps
```

Nonexistent custom properties:

Custom properties that don't exist evaluate to empty.

For example, if the current card doesn't have a custom property called `astCall`, the following statement empties the field:

```
put the astCall of this card into field "Date" -- empty
```

Note: Referring to a nonexistent custom property does not cause a script error. This means that if you misspell a custom property name in a handler, you won't get an error message, so you might not notice the problem right away.

Finding out whether a custom property exists:

The `customKeys` property of an object lists the object's custom properties, one per line:

```
put the customKeys of button 1 into field "Custom Props"
```

To find out whether a custom property for an object exists, you check whether it's listed in the object's `customKeys`. The following statement checks whether a player has a custom property called `doTellAll`:

```
if "doTellAll" is among the lines of the customKeys
```

of player "My Player" then...

You can also look in the Custom Properties pane of the object's property inspector, which lists the custom properties.

The content of a custom property

You set the value of a custom property by using its property name with the set command, in the same way you set built-in properties:

```
set the myCustomProperty of button 1 to false
```

You can see and change all of an object's custom properties in the Custom Properties pane of the object's property inspector: click the custom property you want to change, then enter the new value.

Changing a part of a property:

Like built-in properties, custom properties are not containers, so you cannot use a chunk expression to change a part of the custom property.

Instead, you put the property's value into a variable and change the variable, then set the custom property back to the new variable contents:

```
put the lastCall of this card into myVar
put "March" into word 3 of myVar
set the lastCall of thisCard to myVar
```

Converting text between platforms:

When you move a stack developed on a Mac OS or OS X system to a Windows or Unix system (or vice versa), Revolution automatically translates text in fields and scripts into the appropriate character set. However, text in custom properties is not converted between the ISO and Macintosh character sets. (This is because custom properties can contain binary data as well as text, and converting them would garble the data.)

Characters whose ASCII value is between 128 and 255, such as curved quotes and accented characters, do not have the same ASCII value in the Mac OS character set and the ISO 8859-1 character set used on Unix and Windows systems. If such a character is in a field, it is automatically translated, but it's not translated if it's in a custom property.

Because of this, if your stack displays custom properties to the user—for example, if the stack puts a custom property into a field—and if the text contains special characters, it may be displayed incorrectly if you move the stack between platforms. To avoid this problem, use one of these methods:

ò Before displaying the custom property, convert it to the appropriate character set using the `macToISO` or `ISOToMac` function. The following example shows how to convert a custom property that was created on a Mac OS system, when the property is displayed on a Unix or Windows system:

```
if the platform is "MacOS" then
  answer the myPrompt of button 1
else
```

```
    answer macToISO(the myPrompt of button 1)
end if
```

ò Instead of storing the custom property as text, store it as HTML, using the HTMLText property of fields:

```
set the myProp of this card to the HTMLText of field 1
```

Because the HTMLText property encodes special characters as entities, this ensures that the custom property does not contain any special characters—only the platform-independent encodings for them. You can then set a field's HTMLText to the contents of the custom property to display it:

```
set the HTMLText of field "Display"
to the myProp of this card
```

Storing a file in a custom property

You can use a URL to store a file's content in a custom property:

```
set the myStoredFile of stack "My Stack"
to URL "binfile:mypicture.jpg"
```

You restore the file by putting the custom property's value into a URL:

```
put the myStoredFile of stack "My Stack"
into URL "binfile:mypicture.jpg"
```

Because a custom property can hold any kind of data, you can store either text files or binary files in a custom property. You can use this capability to bundle media files or other files in your stack.

Note: Many Mac OS and OS X files have a resource fork. To store and restore such a file, you can use the resfile URL scheme to store the content of the resource fork separately.

Tip: To save space, compress the file before storing it:

```
set the myStoredFile of stack "My Stack"
to compress(URL "binfile:mypicture.jpg")
```

When restoring the file, decompress it first:

```
put uncompress(the myStoredFile of stack "My Stack")
into URL "binfile:mypicture.jpg"
```

For more information about using URL containers, see the topic [About using URLs, uploading, and downloading](#).

Deleting a custom property

As described above, the `customKeys` property of an object is a list of the object's custom properties. You can set the `customKeys` of an object to control which custom properties it has.

In Transcript, there is no command to delete a custom property. Instead, you place all the custom property names in a variable, delete the one you don't want from that variable, and set the object's `customKeys` back to the modified contents of the variable. This removes the custom property whose name you deleted.

For example, the following statements delete a custom property called `propertyToRemove` from the button `My Button`:

```
get the customKeys of button "My Button"  
set the wholeMatches to true  
delete line lineOffset("propertyToRemove",it) of it  
set the customKeys of button "My Button" to it
```

You can also delete a custom property in the Custom Properties pane of the object's property inspector. Select the property's name and click the Delete button to remove it.

Organizing Custom Properties

Custom properties can be organized into custom property sets. A custom property set is a group of custom properties that has a name you specify.

When you refer to a custom property, Revolution looks for that property in the object's currently-active custom property set. When you create or set a custom property, Revolution creates it in the currently-active custom property set, or sets the value of that property in the currently-active set. One custom property set is active at any one time, but you can use array notation to get or set custom properties in sets other than the current set.

The examples shown so far in this topic assume that you haven't created any custom property sets. If you create a custom property without creating a custom property set for it, as shown in the previous examples, the new custom property becomes part of the object's default custom property set.

Creating custom property sets

To make a custom property set active, you set the object's `customPropertySet` property to the set you want to use. As with custom properties and local variables, if the custom property set you specify doesn't exist, Revolution automatically creates it, so you can create a custom property set for an object by simply switching to that set.

The following statement creates a custom property set called `Alternate` for an object, and makes it the active set:

```
set the customPropertySet of the target to "Alternate"
```

The statement above creates the custom property set.

You can also view, create, and delete custom property sets in the Custom pane of the object's property inspector.

Tip: You can list all the custom property sets of an object using its `customPropertySets` property.

Objects only:

As with custom properties, you can create custom property sets for any object. But you can't create global custom property sets, or custom property sets for a chunk of a field.

Important! Each object can have its own custom property sets, and custom property sets are not shared between objects. Switching one object to a certain custom property set does not switch other objects to using that set.

Custom property set names:

The names of custom property sets should consist of a single word, with any combination of letters, digits, and underscores (`_`). The first character should be either a letter or an underscore.

It is possible to create a custom property set with a name that has more than one word, or that otherwise doesn't conform to these guidelines. However, this is not recommended, because such a custom property set can't be used with the array notation described below.

Note: When you use the Custom Properties pane in the property inspector to create a custom property set, the pane restricts you to these guidelines.

Referring to custom property sets

To switch the active custom property set, set the object's `customPropertySet` property to the name of the set you want to use:

```
set the customPropertySet of button 3 to "Spanish"
```

Any references to custom property refer to the current custom property set. For example, suppose you have two custom property sets named `Spanish` and `French`, and the French set includes a custom property called `Paris` while the Spanish set does not. If you switch to the Spanish set, the `customKeys` of the object does not include `Paris`, because the current custom property set doesn't include that property.

Similarly, the `customProperties` property of an object includes only the custom properties that are in the current custom property set. To specify the `customProperties` of a particular custom property set, you include the set's name in square brackets:

```
put the customProperties[mySet] of this card into myArrayVar
```

If you refer to a custom property that isn't in the current set, the reference evaluates to empty. If you set a custom property that isn't in the current set, the custom property is created in the set. You can have two custom properties with the same name in different custom property sets, and they don't affect each other: changing one does not change the other.

Finding out whether a custom property set exists:

The `customPropertySets` property of an object lists the object's custom property sets, one per line:

answer the customPropertySets of field "My Field"

To find out whether a custom property set for an object exists, you check whether itÆs listed in the objectÆs customPropertySets. The following statement checks whether an image has a custom property set called ôSpanishö:

```
if "Spanish" is among the lines of the customPropertySets  
    of image ID 23945 then...
```

You can also look in the Custom Properties pane of the objectÆs property inspector, which lists the custom property sets in the ôSetö menu halfway down the pane.

The default custom property set:

An objectÆs default custom property set is the set thatÆs active if you havenÆt used the customPropertySet property to switch to another set. Every object has a default custom property set; you donÆt need to create it.

If you create a custom property without first switching to a custom property set—as in the earlier examples in this topic—the custom property is created in the default set. If you donÆt set the customPropertySet property, all your custom properties are created in the default set.

The default custom property set has no name of its own, and is not listed in the objectÆs customPropertySets property. To switch from another set to the default set, you set the objectÆs customPropertySet to empty:

```
set the customPropertySet of the target to empty
```

Using multiple custom property sets

Since only one custom property set can be active at a time, you can create separate custom properties with the same name but different values in different sets. Which value you get depends on which custom property set is currently active.

A translation example:

Suppose your stack uses several custom properties that hold strings in English, to be displayed to the user by various commands. Your stack might contain a statement such as this:

```
answer the standardErrorPrompt of this stack
```

The statement above displays the contents of the custom property called ôstandardErrorPromptö in a dialog box.

Suppose you decide you want to translate your application into French. To do this, you make your original set of English custom properties into a custom property set (which you might call ômyEnglishStringsö), and create a new set called ômyFrenchStringsö to hold the translated properties.

Each set has the same-named properties, but the values in one set are in French and the other in English. You switch between the sets depending on what language the user chooses. The statement:

answer the standardErrorPrompt of this stack

provides either the English or French, depending on which custom property set is active:
ômyEnglishStringsö or ômyFrenchStringsö.

Copying custom properties between property sets:

When itÆs created, a custom property set is empty: that is, there arenÆt any custom properties in it. You put custom properties into a new custom property set by creating the custom properties while the set is active:

```
-- create new set and make it active:  
set the customPropertySet of button 1 to "MyNewSet"  
-- now create a new custom property in the current set:  
set the myCustomProp of button 1 to true
```

You can also use the customProperties property (which was discussed earlier in this topic) to copy custom properties between sets. For example, suppose you have created a full set of custom properties in a custom property set called ômyEnglishStringsö, and you want to copy them to a new custom property set, ômyFrenchStringsö, so you can translate them easily. The following statements create the new custom property set, then copy all the properties from the old set to the new one:

```
-- create the new set:  
set the customPropertySet of this stack to "myFrenchStrings"  
-- copy the properties in the English set to the new set:  
set the customProperties["myFrenchStrings"] of this stack  
  
to the customProperties["myEnglishStrings"] of this stack
```

Custom property sets in the development environment:

The Revolution development environment uses custom properties to create much of its own user interface. All these properties are stored in custom property sets whose names start with ôcRevö.

If youÆre creating custom properties, but not organizing them into your own custom property sets, you donÆt need to worry about RevolutionÆs custom properties. Unless you change an objectÆs customPropertySet, your handlers will never interact with them.

However, if you are using custom property sets—for example, if you have a repeat loop that goes through all of an objectÆs custom property sets—be sure to skip any whose names start with ôcRevö. This ensures that you wonÆt accidentally interfere with any of RevolutionÆs reserved custom properties.

Arrays, custom properties, and custom property sets

All the custom properties in a custom property set form an array. The arrayÆs name is the custom property set name, and the elements of the array are the individual custom properties in that custom property set.

Referring to custom properties using array notation:

You can use array notation to refer to custom properties in any custom property set. This lets you get and set any custom property, even if itÆs not in the current set, without changing the current set.

For example, suppose a button has a custom property named `myProp` which is in a custom property set called `mySet`. If `mySet` is the current set, you can refer to the `myProp` property like this:

```
get the myProp of button 1
set the myProp of the target to 20
```

But you can also use array notation to refer to the `myProp` property, even if `mySet` is not the current set. To refer to this custom property regardless of which custom property set is active, use statements like the following:

```
get the mySet["myProp"] of button 1
set the mySet["myProp"] of the target to 20
```

Note: Because the default custom property set has no name, you cannot use array notation to refer to a custom property in the default set.

Storing an array in a custom property set:

It is not possible to store an array in a single custom property, because each custom property set is already an array of the custom properties in the set.

However, if you store a set of custom properties in a custom property set, the set can be used just like an array. You can think of the custom property set as though it were a single custom property, and the properties in the set as the individual elements of the array.

To store an array variable as a custom property set, use a statement like the following:

```
set the customProperties["myProperty"] of me to theArray
```

The statement above creates a custom property set called `myProperty`, and stores each element in `theArray` as a custom property in the new set. To retrieve a single element of the array, use a statement like this:

```
get the myProperty["myElement"] of field "Example"
```

Deleting a custom property set

As described above, the `customPropertySets` property of an object is a list of the object's custom property sets. You can set the `customPropertySets` of an object to control which custom property sets it has.

In Transcript, there is no command to delete a custom property set. Instead, you place all the custom property set names in a variable, delete the one you don't want from that variable, and set the `customPropertySets` back to the modified contents of the variable. This removes the custom property set whose name you deleted.

For example, the following statements delete a custom property set called `mySet` from the button `My Button`:

```
get the customPropertySets of button "My Button"
```

set the wholeMatches to true
delete line lineOffset("mySet",it) of it
set the customPropertySets of button "My Button" to it

You can also delete a custom property set in the Custom Properties pane of the object's property inspector. Select the set's name from the Set menu, then click the Delete button to remove it.

Attaching Handlers to Custom Properties

When you change a custom property, Revolution sends a setProp trigger to the object whose property is being changed. You can write a setProp handler to trap this trigger and respond to the attempt to change the property. Like a message, this trigger uses the message path, so you can place the setProp handler anywhere in the object's message path.

Similarly, when you get the value of a custom property, Revolution sends a getProp call to the object whose property is being queried. You can write a getProp handler to reply to the request for information. Like a function call, the getProp call also traverses the message path.

Using getProp and setProp handlers, you can:

- ò validate a custom property's value before setting it
- ò report a custom property's value in a format other than what it's stored as
- ò ensure the integrity of a collection of properties by setting them all at once
- ò change an object's behavior when a custom property is changed

Note: setProp triggers and getProp calls are not sent when a built-in property is changed or accessed. They apply only to custom properties.

Responding to changing a custom property

When you use the set command to change a custom property, Revolution sends a setProp trigger to the object whose property is being changed.

A setProp trigger acts very much like a message does. It is sent to a particular object. If that object's script contains a setProp handler for the property, the handler is executed; otherwise, the trigger travels along the message path until it finds a handler for the property. If it reaches the end of the message path without being trapped, the setProp trigger sets the custom property to its new value.

(For more information about the message path, see the topic "About messages and the message path".)

You can include as many setProp handlers in a script as you need. (If a script contains two setProp handlers for the same property, the first one is used.)

The structure of a setProp handler:

A setProp handler begins with the word `setProp`, as a message handler begins with the word `on`. This is followed by the handler's name (which is the same as the name of the custom property) and a parameter that holds the property's new value. A setProp handler, like all handlers, ends with the word `end` followed by the handler's name.

The following example shows a setProp handler for a custom property named `percentUsed`, and can be placed in the script of the object whose custom property it is:

```

setProp percentUsed newAmount
-- responds to setting the percentUsed property
if newAmount is not a number

    or newAmount < zero or newAmount > 100 then
    beep 2
    exit percentUsed
end if
pass percentUsed
end percentUsed

```

When you set the `percentUsed` custom property, the `percentUsed` handler is executed:

```

set the percentUsed of scrollbar "Progress" to 90

```

When this statement is executed, Revolution sends a `setProp` trigger to the scrollbar. The new value of 90 is placed in the `newAmount` parameter. The handler makes sure that the new value is in the range 0–100; if not, it beeps and exits the handler, preventing the property from being set.

Tip: For more information about `setProp` handlers, see the entry for the `setProp` control structure in the Transcript Dictionary.

Passing the `setProp` trigger:

When the `setProp` trigger reaches the engine—the last stop in the message path—the custom property is set. If the trigger is trapped and doesn't reach the engine, the custom property is not set.

To let a trigger pass further along the message path, use the `pass` control structure. The `pass` control structure stops the current handler and sends the trigger on to the next object in the message path, just as though the object didn't have a handler for the custom property.

In the `percentUsed` handler above, if the `newAmount` is out of range, the handler uses the `exit` control structure to halt; otherwise, it executes the `pass` control structure. If the `newAmount` is in the right range, the `pass` control structure lets the property be set. Otherwise, since the trigger is not passed, it never reaches the engine, so the property is not changed.

You can use this capability to check the value of any custom property before allowing it to be set. For example, if a custom property is supposed to be boolean (true or false), a `setProp` handler can trap the trigger if the value is anything but true or false:

```

setProp myBoolean newValue
if newValue is true or newValue is false
then pass myBoolean
exit myBoolean

```

Using the message path with a `setProp` trigger:

Because `setProp` triggers use the message path, a single object can receive the `setProp` triggers for all the objects it owns. For example, `setProp` triggers for all controls on a card are sent to the card, if the `control's` script has no handler for that property. You can take advantage of the message path to implement the same `setProp` behavior for objects that all have the same custom property.

Caution! If a setProp handler sets its custom property, for an object that has that setProp handler in its message path, a runaway recursion will result. To avoid this problem, set the lockMessages property to true before setting the custom property.

For example, suppose that all the cards in your stack have a custom property named `lastChanged`. Instead of putting a setProp handler for this property in the script of each card, you can put a single handler in the stack script:

```
setProp lastChanged newDate
  convert newDate to seconds
  lock messages -- prevent recursion
  set the lastChanged of the target to newDate
  unlock messages
end lastChanged
```

Note: To refer to the object whose property is being set, use the target function. The target refers to the object that first received the setProp trigger—the object whose custom property is being set—even if the handler being executed is in the script of another object.

Setting properties within a setProp handler:

In the `lastChanged` example above, the handler sets the custom property directly, instead of simply passing the setProp trigger. You must use this method if the handler makes a change to the property's value, because the pass control structure simply passes on the original value of the property.

Important! If you use the set command within a setProp handler to set the same custom property for the current object, no setProp trigger is sent to the target object. (This is to avoid runaway recursion, where the setProp handler triggers itself.) Setting a different custom property sends a setProp trigger. So does setting the handler's custom property for an object other than the one whose script contains the setProp handler.

Using this method, you can not only check the new value of a property, and allow it to be set only if it's in range; you can transparently change the value so that it is in the correct range, has the correct format, and so on.

The following example is similar to the `percentUsed` handler above, but instead of beeping if the newAmount is out of range, it forces the new value into the range 0–100:

```
setProp percentUsed newAmount
  set the percentUsed of the target to

    max(zero,min(100,newAmount))
  end percentUsed
```

Nonexistent properties:

If the custom property specified by a setProp handler doesn't exist, the setProp handler is still executed when a handler sets the property. If the handler passes the setProp trigger, the custom property is created.

Custom property sets and setProp handlers:

A setProp handler for a custom property set behaves differently from a setProp handler for a custom property thatÆs in the default set.

When you set a custom property in a custom property set, the setProp trigger is named for the set, not the property. The property name is passed in a parameter. This means that, for custom properties in a set, you write a single setProp handler for the set, rather than one for each individual property.

The following example handles setProp triggers for all custom properties in a custom property set called `myFrenchStrings`, which contains custom properties named `standardErrorPrompt`, `filePrompt`, and perhaps other custom properties:

```
setProp myFrenchStrings[myPropertyName] newValue
-- The myPropertyName parameter contains the name of
-- the property thatÆs being set
switch myPropertyName
case "standardErrorPrompt"
  set the myFrenchStrings["standardErrorPrompt"]

  of the target to return & newValue & return
  exit myFrenchStrings
  break
case "filePrompt"
  set the myFrenchStrings["filePrompt"]

  of the target to return & newValue & return
  exit myFrenchStrings
  break
default
  pass myFrenchStrings
end switch
end myFrenchStrings
```

As you can see from the exit, pass, and end control structures, the name of this setProp handler is the same as the name of the custom property set that it controls—`myFrenchStrings`. Because there is only one handler for all the custom properties in this set, the handler uses the switch control structure to perform a different action for each property that it deals with.

Suppose you change the `standardErrorPrompt` custom property:

```
set the customPropertySet of this stack to "myFrenchStrings"
set the standardErrorPrompt of this stack to field 1
```

Revolution sends a setProp trigger to the stack, which causes the above handler to execute. The property you set—`standardErrorPrompt`—is placed in the `myPropertyName` parameter, and the new value—the contents of field 1—is placed in the `newValue` parameter. The handler executes the case for `standardErrorPrompt`, putting a return character before and after the property before setting it.

If you set a custom property other than `standardErrorPrompt` or `filePrompt` in the `myFrenchStrings` set, the default case is executed. In this case, the pass control structure lets the `setProp` trigger proceed along the message path, and when it reaches the engine, Revolution sets the custom property.

Note: As mentioned above, you can address a custom property in a set either by first switching to that set, or using array notation to specify both set and property. The following example:

```
set the customPropertySet of me to "mySet"  
set the myProperty of me to true
```

is equivalent to:

```
set the mySet["myProperty"] of me to true
```

Regardless of how you set the custom property, if it is a member of a custom property set, the `setProp` trigger has the name of the set—not the custom property itself—and you must use a `setProp` handler in the form described above to trap the `setProp` trigger.

Responding to a request for the value of a custom property

When you use a custom property in an expression, Revolution sends a `getProp` call to the object whose property's value is being requested.

A `getProp` call acts very much like a custom function call. It is sent to a particular object. If that object's script contains a `getProp` handler for the property, the handler is executed, and Revolution substitutes the value it returns for the custom property reference. Otherwise, the call travels along the message path until it finds a handler for the property. If the `getProp` call reaches the end of the message path without being trapped, Revolution substitutes the custom property's value in the expression.

(For more information about the message path, see the topic `About messages and the message path`.)

You can include as many `getProp` handlers in a script as you need. (If a script contains two `getProp` handlers for the same property, the first one is used.)

The structure of a `getProp` handler:

A `getProp` handler begins with the word `getProp`, as a function handler begins with the word `function`. This is followed by the handler's name (which is the same as the name of the custom property). A `getProp` handler, like all handlers, ends with the word `end` followed by the handler's name.

The following example is a `getProp` handler for a custom property named `percentUsed`:

```
getProp percentUsed  
  global lastAccessTime  
  put the seconds into lastAccessTime  
  pass percentUsed  
end lastChanged
```

When you use the `percentUsed` custom property in an expression, the handler is executed:

put the percentUsed of card 1 into myVariable

When this statement is executed, Revolution sends a `getProp` call to the card to retrieve the value of the `percentUsed` property. This executes the `getProp` handler for the property. The example handler stores the current date and time in a global variable before the property is evaluated.

Tip: For more information about `getProp` handlers, see the entry for the `getProp` control structure in the Transcript Dictionary.

Returning a value from a `getProp` handler:

When the `getProp` trigger reaches the engine—the last stop in the message path—Revolution gets the custom property from the object and substitutes its value in the expression where the property was used.

To let a trigger pass further along the message path, use the `pass` control structure. The `pass` control structure stops the current handler and sends the trigger on to the next object in the message path, just as though the object didn't have a handler for the custom property.

To report a value other than the value that's stored in the custom property—for example, if you want to reformat the value first—you use the `return` control structure instead of passing the `getProp` call. The following example is a `getProp` handler for a custom property named `lastChanged`, which holds a date in seconds:

```
getProp lastChanged
  get the lastChanged of the target
  convert it to long date
  return it
end lastChanged
```

The `return` control structure, when used in a `getProp` handler, reports a property value to the handler that requested it. In the above example, the converted date—not the raw property—is what is reported. As you can see from the example, you're not limited to returning the actual, stored value of the custom property. In fact, you can return any value at all from a `getProp` handler.

Important! If you use a custom property's value within the property's `getProp` handler, no `getProp` call is sent to the target object. (This is to avoid runaway recursion, where the `getProp` handler calls itself.)

A handler can either use the `return` control structure to return a value, or use the `pass` control structure to let Revolution get the custom property from the object, but not both, because either of these control structures stops the handler.

If the `getProp` call is trapped before it reaches the engine and no value is returned in the `getProp` handler, the custom property reports a value of empty. In other words, a `getProp` handler must include either a `return` control structure or a `pass` control structure, or its custom property will always be reported as empty.

Using the message path with a `getProp` call:

Because `getProp` calls use the message path, a single object can receive the `getProp` calls for all the objects it owns. For example, `getProp` calls for all controls on a card are sent to the card, if the control's script has no handler for that property. You can take advantage of the message path to implement the same `getProp` behavior for objects that all have the same custom property.

Caution! If a `getProp` handler uses the value of its custom property, for an object that has that `getProp` handler in its message path, a runaway recursion will result. To avoid this problem, set the `lockMessages` property to true before getting the custom property's value.

Nonexistent properties:

If the custom property specified by a `getProp` handler doesn't exist, the `getProp` handler is still executed if the property is used in an expression. Nonexistent properties report empty; getting the value of a custom property that doesn't exist does not cause a script error.

Custom property sets and `getProp` handlers:

A `getProp` handler for a custom property set behaves differently from a `getProp` handler for a custom property that's in the default set.

When you use the value of a custom property in a custom property set, the `getProp` call is named for the set, not the property. The property name is passed in a parameter. This means that, for custom properties in a set, you write a single `getProp` handler for the set, rather than one for each individual property.

The following example handles `getProp` calls for all custom properties in a custom property set called `expertSettings`, which contains custom properties named `fileMenuContents`, `editMenuContents`, and perhaps other custom properties:

```
getProp expertSettings[thePropertyName]
-- The thePropertyName parameter contains the name of
-- the property that's being set
switch thePropertyName
case "fileMenuContents"
    if the expertSettings[fileMenuContents] of the target

        is empty then return "(No items)"
    else pass expertSettings
    break
case "editMenuContents"
    if the expertSettings[editMenuContents] of the target

        is empty then return

        the noviceSettings[editMenuContents] of the target
    else pass expertSettings
    break
default
    pass expertSettings
end switch
end expertSettings
```

As you can see from the pass and end control structures, the name of this getProp handler is the same as the name of the custom property set that it controls—expertSettings. Because there is only one handler for all the custom properties in this set, the handler uses the switch control structure to perform a different action for each property that it deals with.

Suppose you get the fileMenuContents custom property:

```
set the customPropertySet of button 1 to "expertSettings"  
put the fileMenuContents of button 1 into me
```

Revolution sends a getProp call to the button, which causes the above handler to execute. The property you queried—fileMenuContents—is placed in the thePropertyName parameter. The handler executes the case for fileMenuContents: if the property is empty, it returns (No items). Otherwise, the pass control structure lets the getProp call proceed along the message path, and when it reaches the engine, Revolution gets the custom property.

Note: As mentioned above, you can address a custom property in a set either by first switching to that set, or using array notation to specify both set and property. The following example:

```
set the customPropertySet of me to "mySet"  
put the myProperty of me into someVariable
```

is equivalent to:

```
put the mySet["myProperty"] of me into someVariable
```

Regardless of how you specify the custom property, if it is a member of a custom property set, the getProp call has the name of the set—not the custom property itself—and you must use a getProp handler in the form described above to trap the getProp call.

Virtual Properties

A virtual property is a custom property that exists only in a setProp and/or getProp handler, and is never actually set. Virtual properties are never attached to the object. Instead, they act to trigger setProp or getProp handlers that do the actual work.

When you use the set command with a virtual property, its setProp handler is executed, but the setProp trigger is not passed to the engine, and so the property is not attached to the object. When you use a virtual property in an expression, its getProp handler returns a value without referring to the object. In both cases, using the property simply executes a handler.

You can use virtual properties to:

- ò Give an object a set of behaviors
- ò Compute a value for an object
- ò Implement a new property that acts like a built-in property

When to use virtual properties

Because they're not stored with the object, virtual properties are transient: that is, they are re-computed every time you request them. When a custom property depends on other properties that may be set independently, it's appropriate to use a virtual property.

For example, the following handler computes the current position of a scrollbar as a percentage (instead of an absolute number):

```
getProp asPercentage -- of a scrollbar
  put the endValue of the target

  - the startValue of the target into valueExtent
  return the thumbPosition of me * 100 div valueExtent
end asPercentage
```

The `asPercentage` custom property depends on the scrollbar's `thumbPosition`, which can be changed at any time (either by the user or by a handler). Because of this, if we set a custom property for the object, it would have to be re-computed every time the scrollbar is updated in order to stay current. By using a virtual property, you can ensure that the value of the property is never out of date, because the `getProp` handler re-computes it every time you call for the `asPercentage` of the scrollbar.

Virtual properties are also useful solutions when a property's value is large. Because the virtual property isn't stored with the object, it doesn't take up disk space, and only takes up memory when it's computed.

Another reason to use a virtual property is to avoid redundancy. The following handler sets the width of an object, not in pixels, but as a percentage of the object's owner's width:

```
setProp percentWidth newPercentage
  set the width of the target to

    the width of the owner of the target

    * newPercentage div 100
end percentWidth
```

Suppose this handler is placed in the script of a card button in a 320-pixel-wide stack. If you set the button's `percentWidth` to 25, the button's width is set to 80, which is 25% of the card's 320-pixel width. It doesn't make much sense to store an object's `percentWidth`, however, because it's based on the object's width and its owner's width.

Consider using virtual properties whenever you want to define an attribute of an object, but it doesn't make sense to store the attribute with the object because it would be redundant, because possible changes to the object mean it would have to be re-computed anyway, or because the property is too large to be easily stored.

Handlers for a virtual property

As you can see by looking at the example above, a handler for a virtual property is structured like a handler for any other custom property. The only structural difference is that, since the handler has already done everything necessary, there's no need to actually attach the custom property to the object or get its value from the object. When you set a virtual property or use its value, the `setProp` trigger or `getProp` call does not reach the engine, but is trapped by a handler first.

Virtual property setProp handlers:

A setProp handler for an ordinary custom property includes the pass control structure, allowing the setProp trigger to reach the engine and set the custom property (or else it includes a set command that sets the property directly). A handler for a virtual property, on the other hand, does not include the pass control structure, because a virtual property should not be set. Since the property is set automatically when the trigger reaches the end of the message path, a virtual propertyÆs handler does not pass the trigger.

If you examine an objectÆs custom properties after setting a virtual property, youÆll find that the custom property hasnÆt actually been created. This happens because the setProp handler traps the call to set the property; unless you pass the setProp trigger, the property isnÆt passed to Revolution, and the property isnÆt set.

Virtual property getProp handlers:

Similarly, a getProp handler for an ordinary custom property either gets the propertyÆs value directly, or passes the getProp call so that the engine can return the propertyÆs value. But in the case of a virtual property, the object doesnÆt include the property, so getting its value from the object would yield empty.

Creating new object properties

You can use virtual properties to create a new property that applies to all objects, or to all objects of a particular type. Such a property acts like a built-in property, because you can use it for any object. And because a virtual property doesnÆt rely on a custom property being stored in the object, you donÆt need to prepare by creating the property for each new object you create: the virtual property is computed only when you use it in an expression.

The following example describes how to implement a virtual property called `percentWidth` that behaves like a built-in property.

Setting the `percentWidth` property:

Suppose you place the `percentWidth` handler described above in a stack script instead of in a buttonÆs script:

```
setProp percentWidth newPercentage
  set the width of the target to

    the width of the owner of the target

    * newPercentage div 100
end percentWidth
```

Because setProp triggers use the message path, if you set the `percentWidth` of any object in the stack, the stack receives the setProp trigger (unless itÆs trapped by another object first). This means that if the handler is in the stackÆs script, you can set the `percentWidth` property of any object in the stack.

If you place the handler in a backscript, you can set the `percentWidth` of any object, anywhere in the application.

Note: To refer to the object whose property is being set, use the target function. The target refers to the object that first received the setProp trigger—the object whose custom property is being set—even if the handler being executed is in the script of another object.

Getting the `percentWidth` property:

The matching getProp handler, which lets you retrieve the `percentWidth` of an object, looks like this:

```
getProp percentWidth
  return 100 * (the width of the target

    div the width of the owner of the target)
end percentWidth
```

If you place the handler above in a card button's script, the following statement reports the button's width as a percentage:

```
put the percentWidth of button "My Button" into field 12
```

For example, if the stack is 320 pixels wide and the button is 50 pixels wide, the button's width is 15% of the card width, and the statement puts `15` into the field.

Like the setProp handler for this property, the getProp handler should be placed far along the message path. Putting it in a stack script makes the property available to all objects in the stack; putting it in a backscript makes the property available to all objects in the application.

Limiting the `percentWidth` property:

Most built-in properties don't apply to all object types, and you might also want to create a virtual property that only applies to certain types of objects. For example, it's not very useful to get the width of a substack as a percentage of its main stack, or the width of a card as a percentage of the stack's width.

You can limit the property to certain object types by checking the target object's name:

```
setProp percentWidth newPercentage
  if word 1 of the name of the target is "stack"

    or word 1 of the name of the target is "card"

    then exit setProp
  set the width of the target to

    the width of the owner of the target

    * newPercentage div 100
end percentWidth
```

The first word of an object's name is the object type, so the above revised handler ignores setting the `percentWidth` if the object is a card or stack.

Summary

In this topic, you have learned that:

- ò A custom property is a property that you define. You can create custom properties for any object and put any kind of data in them.
- ò You can create, delete, and change an object's custom properties in the Custom Properties pane of the object's property inspector.
- ò You create a custom property by setting it to a value. If the custom property doesn't exist, Revolution creates it.
- ò The customKeys property contains a list of all an object's custom properties. The customProperties property is an array of all an object's custom properties, along with their values.
- ò A custom property set is a collection of custom properties with a name you specify. To make a custom property set active or to create a new custom property set, you set the object's customPropertySet property to the set's name.
- ò To refer to a custom property in a custom property set that's not the active set, you use array notation.
- ò When you change a custom property, Revolution sends a setProp trigger to the object whose property is being changed. When you request a custom property's value, Revolution sends a getProp call to the object.
- ò A virtual property is a custom property that is never attached to the object. Instead, it exists only in the form of a getProp handler and/or a setProp handler, which implement the behavior of a custom property.

About filename specifications and file paths

See Also:

About file types, application signatures, and file ownership, How to change the current folder used for file operations, How to determine whether a file exists, How to determine whether a file path is relative or absolute, How to find out the location of the current stack's file, How to get the location of the user's home directory, How to list the contents of a folder, How to make an alias, symbolic link, or shortcut to a file, Why can't Revolution find a file I specified?, Recipe for converting an absolute path to a relative path, Recipe for converting a relative path to an absolute path, answer file command, answer folder command, ask file command, create alias command, create folder command, defaultFolder property, file keyword, filename property, filename of stack property, files function, folders function, longFilePath function, name property, revMacFromUnixPath function, revUnixFromMacPath function, shortFilePath function, URL keyword

A file path is a way of describing the location of a file or folder so that it can be found by a handler. File paths are used throughout Revolution: when you read to and write from text files, when you reference an external QuickTime file to display in a player, and in many other situations. If your application refers to external files in any way, an understanding of file path syntax is essential.

This topic discusses the syntax for creating and reading a file reference, and how to relate file paths to the location of your application so that they'll be accessible when your application is installed on another system with a different folder structure.

To fully understand this topic, you should have a basic understanding of computer file systems: what a folder is, how to move items from one folder to another, and how to locate a file using your operating system.

Contents:

What is a File Path?

Absolute and Relative Paths

Using File URLs

Special Folders

Summary

What is a File Path?

A file path is a description of the exact location of a file or folder. The file path is created by starting at the top of the computer's file system, naming the disk or volume that the file is on, then naming every folder that encloses the file, in descending order, until the file is reached.

Locating a file

For example, suppose you want to describe the location of a file called "My File", which is located inside a folder called "My Folder". That folder is in turn located inside a folder called "Top Folder", which is on a drive called "Hard Disk". You need all this information to completely describe where the file is:

Hard Disk

Top Folder

My Folder My File

If someone tells you which disk the file is on, then which folder to open, and so on, you can find the file by opening each successive icon on your computer's desktop. By starting with the disk, then opening each enclosing folder in succession until you arrive at the file, you can find exactly the file that's being described.

The structure of a file path

A file path specifies each level of the hierarchy that encloses the file. Revolution presents the information in a file path that might look like this:

```
/Hard Disk/Top Folder/My Folder/My File
```

You can see that to write a file path, you start by naming the disk the file is on, then add each enclosing folder in order until you arrive at the file.

Tip: To see the path to a file, enter the following in the message box:

```
answer file "Choose a file: ";put it
```

This displays the file path for the file you choose. Examine the file paths of several files in different places to get an idea of what they look like.

Each platform has its own way for programmers to specify file paths. The file path shown above is in the usual style for file paths on Unix systems. For cross-platform compatibility, Revolution uses this same style of file path regardless of the current platform. This way, you can specify file paths in your scripts without having to convert them when you switch platforms. All Transcript commands, properties, and functions use the same Unix-style file paths.

Cross-platform note: On Windows systems, disks are named with a drive letter followed by a colon character (:). A typical Revolution file path on a Windows system looks like this:

```
C:/folder/file.txt
```

File paths on OS X systems:

On OS X systems, the startup disk, rather than the desktop, is used as the top level of the folder structure. This means that the startup disk's name does not appear in file paths. Instead, the first part of the file path is the top-level folder that the file is in.

If the disk "Hard Disk" is the startup disk, a typical path on OS X systems might look like this:

```
/Top Folder/My Folder/My File
```

Notice that the disk name isn't part of this path.

Tip: If you need to find out the startup disk's name, check the first disk name returned by the `volumes` function.

For files on a disk that isn't the startup disk, the file path starts with `/Volumes` instead of `/`. A typical file path to a file that's on a non-startup disk on an OS X system looks like this:

```
/Volumes/Swap Disk/Folder/file.txt
```

Folder paths

You construct the path of a folder the same way as the path to a file. A folder path always ends with a slash character (/). This final slash indicates that the path is to a folder rather than a file.

For example, this pathname describes a folder called `Project` inside a folder called `Forbin` on a disk named `Doomsday`:

```
/Doomsday/Forbin/Project/
```

If `Project` is a file, its pathname looks like this, without the final slash:

```
/Doomsday/Forbin/Project
```

File paths for OS X bundles

A bundle is a special type of folder, used on OS X, that is presented to the user as a single file but that is maintained internally by the operating system as a folder. Many OS X applications—including Revolution and the applications it creates—are stored and distributed as bundles that contain several files. When the user double-clicks the bundle, the application starts up, instead of a folder window opening to show the bundle's contents.

You can take advantage of the bundle concept to include any needed support files with your application. If you place the files in the application's bundle, users ordinarily never see them, and the entire application's support files and all behave as a single icon.

Tip: To see the contents of a bundle, Control-click the bundle and choose `Show Package Contents` from the contextual menu.

Most of the time, the distinction between bundles and files doesn't matter. However, there are a few situations where you must take it into account, treating a bundle as a folder.

Referring to bundles:

When using a bundle's file path, you must include a slash at the end of the path to indicate that you're referring to a folder:

```
launch "../My Bundled Application/" -- a bundle
```

Moving, renaming, or deleting a bundle:

When using the rename command, to rename a bundle, use the rename folder form of the command:

```
rename folder "/Volumes/Disk/Applications/MyApp/"  
to "/Volumes/Disk/Applications/OtherApp/"
```

Similarly, when dealing with a bundle, use the delete folder command instead of delete file, and the `revCopyFolder` command instead of `revCopyFile`.

Referring to files inside a bundle:

When referring to a file that's inside a bundle, you can treat the bundle just as if it were a folder. For example, if you have placed a file called `My Support.txt` inside your application's bundle, the absolute path to the file might look like this:

```
/Volumes/Disk/Applications/MyApp/My Support.txt
```

The / character in a file or folder name

The slash (/) is not a legal character in Unix or Windows file or folder names, but it is legal for Mac OS file or folder names to contain a slash. Since a slash in a file or folder name would cause ambiguity—is the slash part of a name, or does it separate one level of the hierarchy from the next?—Revolution substitutes a colon (:) for any slashes in folder or file names on Mac OS systems.

For example, if a file on a Mac OS system is named “Notes from 12/21/93”, you refer to it in a script as “Notes from 12:21:93”. Since the colon is not a legal character in Mac OS folder or file names, this removes the ambiguity.

Absolute and Relative File Paths

When describing how to get to a file, you have two options. You can start from the top level, the name of the disk, and name each of the enclosing folders until you get to the file. (This is called an absolute path, because it is independent of where you start from.) Or you can start from the current folder and describe how to get to the file from there. (This is called a relative path, because it depends on where you start.)

All the file paths shown so far in this topic are absolute paths.

Absolute file paths

An absolute file path starts at the top of the file system, and describes how to get to a folder or file by working from the top down. (On Unix systems, the top of the file system is the root directory. On Mac OS and Windows systems, the top component is the name of a disk.)

Absolute file paths do not depend on which folder your stack file is in or on where the current folder is. An absolute path to a particular folder or file is always written the same way.

For example, suppose your application is in a folder called “Application Folder”, and you want to specify a file called “Westwind” which is in a folder called “Stories” inside “Application Folder”.

```
Hard Disk
  Top Folder
    My Folder
      My File
        Application Folder
          My Application
            Stories
              Westwind
```

The absolute file path of your application looks like this:

/Hard Disk/Application Folder/My Application

and the absolute path of the “Westwind” file looks like this:

/Hard Disk/Application Folder/Stories/Westwind

Cross-platform note: On Mac OS, OS X, and Unix systems, absolute file paths always start with a slash character. On Windows systems, absolute file paths always start with a drive letter followed by a colon (:).

Relative file paths

Now suppose you want to tell someone how to get to the `Westwind` file, starting from the folder containing the application.

Since the application is in `Application Folder`, we don't need to include the steps to get to `Application Folder`. Instead, we can describe the location of the `Westwind` file with this relative pathname:

`Stories/Westwind`

This relative pathname starts at `Application Folder`—the folder that holds the application—and describes how to get to the `Westwind` file from there: you open the folder `Stories`, then find `Westwind` inside it.

A relative file path starts at a particular folder, rather than at the top of the file system like an absolute file path. The relative file path builds a file path from the starting folder to the file or folder whose location is being specified.

Finding the current folder:

By default, the current folder is set to the folder containing the application (either the Revolution development environment or your application, depending on whether your application is a standalone). So in the example above, the current folder is `Application Folder`, because that's where the running application is located.

Tip: To change the current folder, set the `defaultFolder` property.

Going up to the parent folder:

What if the folder you want to specify is not inside the current folder? For example, suppose you want to specify the folder `Top Folder` using a relative path?

The relative path `../` indicates the current folder's parent folder. If the current folder is `Stories`, the relative path

`../`
means the same thing as the absolute path
`/Hard Disk/Application Folder/`

To move up a level, add `../` to the start of the path. For example, suppose the absolute path to `My File` looks like this:

`/Hard Disk/Top Folder/My Folder/My File`

`Application Folder` is another folder on the hard disk. To describe the location of `My File` starting from `Application Folder`, use this relative path:

`../Top Folder/My Folder/My File`

As before, this path starts from `Application Folder`, where the application is located. The `../` part of the relative path moves up one level to `Hard Disk`. Since the disk contains `Top Folder`, the rest of the file path is just like the absolute path.

Going up multiple levels:

To go up more than one level, use more than one `../`. To go up two levels, use `../../`; to go up three levels, use `../../..`, and so forth.

For example, suppose the current folder is `Stories`, and its absolute path looks like this:
`/Hard Disk/Application Folder/Stories/`

To get to `My Application` in `Application Folder`, you go up one level to `Application Folder`, then down one level to `My Application`. The relative path looks like this:
`../My Application`

To get to `Top Folder` on `Hard Disk`, you go up two levels to `Application Folder`, then to `Hard Disk` and then down one level to `Top Folder`. The relative path looks like this:
`../../Top Folder/`

Starting at the home directory:

On OS X and Unix systems, the `~` character designates a user's home directory.

A path that starts with `~/` is a relative path starting with the current user's home directory. A path that starts with `~/`, followed by the user ID of a user on that system, is a relative path starting with that user's home directory.

When to use relative and absolute file paths

Absolute file paths and relative file paths are interchangeable, as long as they both lead to the same file or folder. You can use either type of path in any Transcript command, function, or property that requires a file path. Which one to use depends on a number of factors.

Absolute file paths are easy to understand, and they don't change depending on the current folder. This can be an advantage if you are changing the default folder frequently.

Relative file paths are a bit more complicated to understand than absolute paths, but they are more flexible, since they don't require you to specify the entire path to the file or folder. If you rename a disk or an enclosing folder, or move your entire application to another system with different file and folder names, the relative file path from your stack file will still be correct, while the absolute file path won't be.

In the examples given above, if you change the disk name `Hard Disk`, the absolute paths is no longer correct, but the relative path is still correct because it doesn't need to specify any name in the folder hierarchy above where you're working. If you copy the entire `Application Folder` to another system with a different folder structure, all the relative pathnames to files and folders inside `Application Folder` are still correct, so long as you don't change anything within `Application Folder`.

If you're using a file path that was created on the same system—for example, if you're using a file path returned by the `answer file` command to write to a file—this doesn't matter, but if you're storing a file path in your stack for later use, a relative path is more reliable.

Because relative paths don't depend on your hard disk name or folder structure, you should use relative paths to designate files or folders that you specify when creating the application. For example, if your application opens a text file that's kept in the same folder as the application, it's better to refer to it using a relative path.

It's OK to use absolute paths to specify files or folders that the user selects after installation. For example, if you ask the user to select a file (using the `answer file` command) and read data from the file, there's no need to convert the absolute path that the `answer file` command provides to a relative path. Because you're using the path right after you get it from the `answer` command, you know that the disk name and folder structure aren't going to change between getting the path and using it.

Using File URLs

You can refer to the contents of files using the `file`, `binfile`, or `resfile` URL types. These three URL types use the same file path syntax described above, and can be combined with file paths to construct a valid URL.

For example, the following statement gets the contents of the file `My File`, using the absolute path that was given as an example at the start of this topic:

```
get URL "file:/Hard Disk/Top Folder/My Folder/My File"
```

The URL is constructed simply by adding `file:` to the beginning of the absolute path.

You can also use a URL with a relative path, using the rules discussed above:

```
put URL "file:Stories/Westwind" into field "Current"  
put myStuff into URL "file:../Top Folder/My Folder/My File"
```

To create a URL from a file path that Revolution provides, use the `&` operator:

```
answer file "Please choose a file to get:"  
get URL ("file:" & it)
```

You can create `binfile` and `resfile` URLs in the same way:

```
put the storedFile of me into URL "binfile:../file.mpg"  
put URL ("resfile:" & thePath) into myResources
```

Special Folders

Mac OS, OS X, and Windows systems each have a set of special-purpose folders. For example, the contents of the desktop reside in a special folder; there is a folder set aside for fonts; there is a folder for application preferences; and so on.

These special folders don't always have the same name and location, so you can't rely on a stored file path to locate them. (For example, on Mac OS systems, the System Folder can be placed anywhere on the hard drive and given any name. And some Windows special folders are named or placed differently depending on what version of Windows is running.)

To find out the name and location of a special folder, regardless of its exact location or the operating system version, you use the `specialFolderPath` function. The function supports a number of forms for each operating system, describing the special folders for each one. For example, the following statement gets the path to the Preferences folder on a Mac OS system:

```
put specialFolderPath("Preferences") into myPath
```

And the following statement gets the path to the Start menu's folder on a Windows system:

```
put specialFolderPath("Start") into myPath
```

(For a complete list of possible folders, see the entry in the Transcript Dictionary for the `specialFolderPath` function.)

Additional Mac OS and OS X folders:

For Mac OS and OS X systems, you can also specify a four-character special folder constant that corresponds to a particular special folder. Some constants are meaningful only on Mac OS or only on OS X, but not both, so check any constant before using it in a handler.

Here are a few useful special folders:

```
ò get specialFolderPath("asup") -- Application Support
```

The Application Support folder holds support files, such as code libraries, for a particular application. If you want to store such files outside your application's own folder, create a folder inside the Application Support folder for your application and put the files there.

```
ò get specialFolderPath("trsh") -- Trash can
```

The Trash folder holds any files and folders that are in the user's trash can. If you want to delete a file without making the deletion undoable, move the file to the Trash.

```
ò get specialFolderPath("docs") -- Documents
```

The Documents folder holds the user's document files. This is a good place to use as the default folder when offering to save a user's files. (Mac OS systems have the Documents folder installed by default, but the user often removes it, so don't count on the folder being present on Mac OS systems.)

You can find a list of available constants on the Apple web site at

<http://developer.apple.com/techpubs/macosx/Carbon/Files/FolderManager/Folder_Manager/folder_manager_ref/constant_6.html#apple_ref/c/tdef/FolderType>.

Additional Windows folders:

For Windows systems, you can also specify a number constant that corresponds to a particular special folder. Some numbers have meaning on some versions of Windows but not others, so check any constant before using it in a handler.

Here are a few useful special folders:

```
ò get specialFolderPath(26) -- CSIDL_APPDATA (0x001a)
```

The Application Data folder holds user-specific files, such as preference files for your application.

```
ò get specialFolderPath(7) -- CSIDL_STARTUP (0x0007)
```

The Startup folder holds programs that are executed whenever the user starts up Windows (or logs in, on newer Windows versions).

```
ò get specialFolderPath(11) -- CSIDL_STARTMENU (0x000b)
```


The Start Menu folder holds the contents of the Windows Start button.

You can find a list of available numbers on the Microsoft web site at
<<http://msdn.microsoft.com/library/en-us/shellcc/platform/shell/reference/enums/csidl.asp>>.

Note: The CSIDL numbers are provided in hexadecimal notation, so youÆll need to translate them into decimal numbers to use them with the specialFolderPath function. For example, the CSIDL number provided for the òMy Musicö special folder is hexadecimal 0x000D. The hexadecimal number 000D is equal to the decimal number 13, so to get the path to the òMy Musicö special folder, you use the expression specialFolderPath(13).

Summary

In this topic, you have learned that:

- ò A file path is a description of the exact location of a file or folder, with slashes (/) separating the components of the path.

- ò Folder paths end with a slash (/).

- ò An absolute file path begins at the top of the folder structure, usually with a disk name, and includes all the folders that enclose the folder or file youÆre locating. Windows paths begin with a drive letter and colon. OS X paths begin with ò/Volumesö.

- ò A relative file path begins at the defaultFolder and moves up, down, or sideways in the folder structure as needed to reach the folder or file youÆre locating.

- ò URLs for local files (file, binfile, or resfile URLs) follow the same rules as file paths.

- ò You use the specialFolderPath function to get the file path of folders that are used for special purposes by the operating system.

About file types, application signatures, and file ownership

See Also:

About filename specifications and file paths, How to assign an icon to a Mac OS standalone application, How to assign an icon to an OS X standalone application, How to assign an icon to a Windows standalone application, How to associate files with a Mac OS standalone application, How to associate files with an OS X standalone application, How to associate files with a Windows standalone application, How to copy a resource fork, How to respond to double-clicking a file for Mac OS or OS X, How to respond to double-clicking a file for Windows, How to store preferences or data for a standalone application, Recipe for creating a file with a custom extension or type

When you double-click a document file, it automatically opens in the application itÆs associated with. Each operating system has a different method for associating files with an application. In order to create files that belong to your standalone application, you need to set up the association appropriately for each platform you distribute on.

This topic describes how to correctly associate your application with the files it creates.

To fully understand this topic, you should have a basic understanding of computer file systems: what a folder is, how to move items from one folder to another, and how to locate a file using your operating system.

Contents:

Mac OS File Types and Creators

OS X File Types and Creators

Windows File Extensions and Ownership

Unix File Extensions

Summary

Mac OS File Types and Creators

When a file is saved on a Mac OS system, a four-character creator signature is saved with it. The creator signature specifies which application owns the file. Every Mac OS and OS X application should have a unique creator signature. (Apple maintains a registry of creator signatures on its web site at

<<http://developer.apple.com/dev/cftype/>>.)

Mac OS files have a separate file type, also four characters long, which specifies the format of the file. The file type is also used to determine which applications (other than the owner) can work with the file. The file type of all applications is ôAPPLö.

Applications that donÆt own files

To assign your unique creator signature when building an application, enter the signature on the Mac OS tab in Step 3 of the Distribution Builder. Revolution automatically includes the resources needed for Mac OS to recognize the creator signature.

Applications that own their own files

If your application creates files with your application's creator signature, you should include in your application a set of resources for each file type you use. Once you have built your application with the Distribution Builder, open the application file in ResEdit and follow these steps:

1. Open the BNDL resource window, then open the BNDL 128 resource. The BNDL 128 resource contains a single entry (APPLö).
2. Choose "Create New File Type" from the Resources menu. A new entry appears below the APPLö entry.
3. In the Type field, enter the four-character file type. If the format for this type of file is standard (such as plain text), use a standard type (such as TEXTö). If the format belongs to your application, use a custom file type of your choice.

Important! Apple reserves all file types with no uppercase letters. If you use a custom file type for your application, make sure it contains at least one uppercase letter.

Repeat steps 2-3 for each different file type your application can create.

When your application creates files, set the fileType property to the desired creator signature and file type for the new file. (For stack files created with the save command, use the stackFileType property instead.) When creating files, the application uses the current value of the fileType or stackFileType property to determine what creator and file type the new file should have.

It's important to understand that a file's creator signature determines which application is launched automatically when you double-click the file, but doesn't prevent other applications from being able to open that file. For example, if your application creates files of type TEXTö, any text editor can open the files. If your application creates stack files, and uses the file type RSTKö, then Revolution will be able to open the stack files, as well as your application.

Installing custom icons

Each Mac OS file may display any of six different icons, depending on context and on the number of colors the screen can display: large (32x32 pixel) icons and small (16x16 pixel) icons, each in black-and-white, 16 colors, and 256 colors.

Mac OS provides default icons that are used for applications and documents that don't have their own. If you want your application or the documents it owns to display a custom icon, you must create the icons and then attach them to the application.

Custom application icons:

If you want to include a custom icon for your application, use ResEdit or a similar tool to create a set of icon resources. There are six standard icon resource types: ICN# (black-and-white), icl4 (four-bit color), icl8 (8-bit color), ics# (black-and-white small), ics4 (4-bit small), and ics8 (8-bit small). Each of these application icons should have the resource ID 128.

Save the icons in a single file, and use the "Include resources from file" option on the Mac OS tab in Step 3 of the Distribution Builder to specify the file. When you build the application, the icons will be included in the application's file.

Note: You can also copy your icon to the clipboard and paste it into the application's Get Info window. The application file will display the icon you paste. However, this is not as good a solution, because the user use the Get Info window to remove the icon. And since this technique does not include small icons, the operating system scales down the full-sized icon when a small icon is needed, which may cause your icon to look blurred or distorted.

Custom file icons:

The Mac OS Finder uses the file type along with the creator to locate a file's icon. To give each file type its own custom icon, create the six standard icon resources (ICN#, icl4, icl8, ics#, ics4, and ics8) discussed above, a complete set of six for each file type. The first file type's icons should have the resource ID 129, the second should have the resource ID 130, and so on.

Store these icon resources in the same file as the application icons, so that they're included automatically when you build the application. (Or, if you prefer, you can store them separately and copy them into the application file using ResEdit.)

To link each icon with its file type, add a fourth step to the procedure described earlier for adding file types to an application:

4. Click the Finder Icons field and choose the ICN# icon for the file type.

Important! For the correct icon to appear on files your application creates, the file's creator signature and file type must be correct and correspond exactly to the creator signature and file type in the application's BNDL resource. Both creator signatures and file types are case-sensitive, so uppercase letters are not equivalent to lowercase letters. Set the fileType property before creating the file in order to give it the correct creator signature and file type.

File extensions

You can add an extension to the name of any Mac OS file. In general, the operating system pays no attention to the extension; it's simply part of the file's name.

You may want to add an extension if, for example, the file is going to be moved between systems and you want to ensure that it will be handled correctly if moved to a Windows system.

OS X File Types and Creators

Creator signatures and file types work on OS X systems much the same way as on Mac OS systems. Each file has a unique four-character creator signature and a four-character file type, which determine which application owns the file.

While Mac OS applications store information about an application's creator signature and file types as resources, OS X applications instead store this information in a property list file, or plist. An application's plist is stored as part of the application bundle.

Applications that don't own files

To assign your unique creator signature when building an application, enter the signature on the OS X tab in Step 3 of the Distribution Builder. Revolution automatically includes the creator signature in the application's plist.

Applications that own their own files

If your application creates files with your application's creator signature, you should include in your application's plist an entry for each file type you use. Once you have built your application with the Distribution Builder, follow these steps to open the plist file:

1. Control-click your application's icon and choose "Show Package Contents" from the contextual menu.
2. Open the "Contents" folder inside the application bundle.
3. Open the "Info.plist" file inside the "Contents" folder.

At this point, you have two choices. If you have installed Apple's developer tools, you have an application called "Property List Editor", which you can use to make changes to the plist file. Otherwise, you can edit the file in a text editor.

4. Locate the information for the document type:

In Property List Editor, expand the "Root" node, then expand the "CFBundleDocumentTypes" node, then expand the "" node.

In a text editor, locate "CFBundleDocumentTypes". Below it, note the tags "<array>" and "<dict>". The information for the first document type is between "<dict>" and "</dict>".

5. Enter the file description, which is a short phrase describing what kind of file this is:

In Property List Editor, change the value of "CFBundleTypeName" to the description you want to use.

In a text editor, locate "CFBundleTypeName" in the document information. Below it is the file description, enclosed between "<string>" and "</string>":

```
<string>Revolution Stack</string>
```

Change the description to the one you want to use.

Important! Do not change the tags (enclosed in "<" and ">"). Only change what's between them.

6. Enter the file extension:

In Property List Editor, expand "CFBundleTypeExtensions" and enter the file extension in the "" node.

In a text editor, locate "CFBundleTypeExtensions" in the document information. Below it is the extension, enclosed in "<array>" and "<string>" tags. Change the extension to the one you want to use.

7. Enter the four-character file type:

In Property List Editor, expand "CFBundleTypeOSTypes" and enter the file type in the "" node.

In a text editor, locate "CFBundleTypeOSTypes" in the document information. Below it is the file type, enclosed in "<array>" and "<string>" tags. Change the file type to the one you want to use.

If the format for this type of file is standard (such as plain text), use a standard type (such as `TEXT`). If the format belongs to your application, use a custom file type of your choice.

Important! Apple reserves all file types with no uppercase letters. If you use a custom file type for your application, make sure it contains at least one uppercase letter.

If you want to assign more file types to your application, copy the portion of the plist file inside the `CFBundleTypes` array between `<dict>` and `</dict>`, including these tags. The `CFBundleTypes` node should now contain two `<dict>` nodes and all their contents. Repeat steps 5-7 for each different file type your application can create.

When your application creates files, set the `fileType` property to the desired creator signature and file type for the new file. (For stack files created with the `save` command, use the `stackFileType` property instead.) When creating files, the application uses the current value of the `fileType` or `stackFileType` property to determine what creator and file type the new file should have.

It's important to understand that a file's creator signature determines which application is launched automatically when you double-click the file, but doesn't prevent other applications from being able to open that file. For example, if your application creates files of type `TEXT`, any text editor can open the files. If your application creates stack files, and uses the file type `RSTK`, then Revolution will be able to open the stack files, as well as your application.

Installing custom icons

Each OS X file displays an icon. This icon is scalable, so it can be used at any size the operating system requires. If you want your application or the documents it owns to display a custom icon, you must create the icons and attach them to the application.

Custom application icons:

If you want to include a custom icon for your application, use Icon Composer (on the Apple Development Tools CD) or a similar utility to create the icon file.

Save the icons in a single file, and use the `Application Icon` option on the OS X tab in Step 3 of the Distribution Builder to specify the file. When you build the application, the icon will be included in the application bundle.

Custom file icons:

The OS X Finder uses the file type along with the creator to locate a file's icon. To give each file type its own custom icon, create the icon and save it to a file. Then add it to the application bundle:

1. Control-click your application's icon and choose `Show Package Contents` from the contextual menu.
2. Open the `Contents` folder inside the application bundle.
3. Open the `Resources` folder inside the `Contents` folder. You'll see a file called `RevolutionDoc.icns`. Replace this file with your icon file.

If you want to call the file something other than `RevolutionDoc.icns`, you must modify the application's plist file. In the plist, look for `CFBundleTypeIconFile`. Inside it (using Property List

Editor) or beneath it (using a text editor) is the file name `ôRevolutionDoc.icnsö`. Replace this with the file name you want to use for your document icon.

Important! For the correct icon to appear on files your application creates, the file's creator signature and file type must be correct and correspond exactly to the creator signature and file type in the application's BNDL resource. Both creator signatures and file types are case-sensitive, so uppercase letters are not equivalent to lowercase letters. Set the fileType property before creating the file in order to give it the correct creator signature and file type.

File extensions

You can add an extension to the name of any OS X file. When the user double-clicks a file with no creator signature, the operating system uses the extension to determine which application to use to open the file.

An application bundle's name should end with the extension `ô.appö`.

Note: Apple's recommendations for determining file type and creator on OS X systems are currently in flux. The recommended method for the present is to set a file type and creator signature, and also attach an extension to the end of each file's name when it is created. Valid extensions on OS X systems are up to twelve characters in length, and may include the letters a-z, the digits 0-9, \$, %, _, or ~. For up-to-date information on Apple's recommendations for OS X, see Apple's developer documentation at <http://www.apple.com/developer/>.

Windows File Extensions and Ownership

When a file is saved on a Windows system, a three-character extension is usually added to the file's name. The extension specifies the format of the file.

To determine which application to launch when the user double-clicks a file, Windows checks the Windows registry to find out what application has registered itself as owning the file's extension. Each application can add keys to the registry to identify certain file extensions as belonging to it.

Applications that don't own files

If your application does not create files that you want the application to own, you don't need to make any modifications to the registry or specify any extensions.

Applications that own their own files

If your application creates files with its own custom extension, when you install the application, you should make changes to the Windows registry to identify the extension as belonging to your application.

Popular Windows installer programs will make these registry changes automatically for you. You can also perform these registry changes using the `setRegistry` function. (For example code, see the topic `ôRecipe for registering a Windows extensionö` in the Transcript Cookbook section of the documentation.)

Installing custom icons

Each Windows file can display its own icon. You can have separate icons for your application and for files it owns.

Windows icons must be in .ico format, 32x32 pixels, four-bit color (16 colors). You can use whatever colors you want, but black must be the first color and white must be the last. (Some Windows versions allow larger icons or icons with more colors, but Revolution does not support those icon formats.)

Custom application icons:

If you want to include a custom icon for your application, use the Application Icon option on the Windows tab in Step 3 of the Distribution Builder to specify the icon file. When you build the application, the icon will be included in the application.

Custom file icons:

To include a custom icon for your documents, use the Document Icon option on the Windows tab in Step 3 of the Distribution Builder to specify the icon file. When you build the application, the icon will be included in the application.

Important! For the correct icon to appear on files your application creates, the file's extension must be registered in the Windows registry.

File extensions

You can add an extension to the name of any Windows file. The extension may contain letters A-Z, digits 0-9, ' (single quote), !, @, #, \$, %, ^, &, (,), -, _, {, }, ` , or ~.

The Windows registry associates applications with the extension for the files they own.

Unix File Extensions

Unix systems do not have an overall required method for specifying a file's type, but most files on a Unix system are created with extensions in the file name, similar to the extensions used on Windows systems. These extensions may be of any length and may include any characters (other than /).

Summary

In this topic, you have learned that:

- ò Mac OS and OS X systems use a four-character creator signature, which is unique to each application, to identify applications and to associate files with the application that owns them. These operating systems use a four-character file type to specify the format of files. OS X also may use an extension to identify the owner of files.

- ò Windows systems use a file extension that is registered in the Windows registry to associate files with the application that owns them.

- ò To set up file associations and to attach custom icons to a Mac OS application and its documents, you include a resource file with the icon or icons in the Distribution Builder. Then you use the application's BNDL resource to associate each file type with its application.

- ò To attach custom icons to an OS X application, you include an icon file in the Distribution Builder. To set up file associations, you modify the application's property list file.

- ò To attach custom icons to a Windows application and its documents, you create the icons in 4-bit 32x32-pixel .ico format and include them in the Distribution Builder.

ò To associate a file created by your Mac OS or OS X application with the application, you set the `stackFileType` property (for stack files) or `fileType` property (for other files) before creating the file.

About colors and color references

See Also:

Color Names Reference, How to make an object the same color as its owner, How to change an object's color, How to change the color of text, How to find out the actual color of an object, How to restore the default colors of an object, Why does an image have the wrong colors?, Recipe for setting the red channel of an object, Recipe for translating a color name to an RGB numeric triplet, Text menu > Color, `accentColor` property, `backdrop` property, `backgroundColor` property, `borderColor` property, `bottomColor` property, `brushColor` property, `colorMap` property, `colorNames` function, `colors` property, `colorWorld` property, `effective` keyword, `focusColor` property, `foregroundColor` property, `hiliteColor` property, `linkColor` property, `linkHiliteColor` property, `linkVisitedColor` property, `lockColorMap` property, `lock color map` command, `mouseColor` function, `patterns` property, `penColor` property, `screenColors` function, `screenDepth` function, `screenType` function, `shadowColor` property, `selectionHandleColor` property, `topColor` property

Each Revolution object has eight color properties, which affect different parts of each object type. You can set each of these properties independently to any color, using either numeric or named colors.

This topic discusses Revolution's color model, the effect of each color on object appearance, and how to construct a specification for a color.

To fully understand this topic, you should know how to create objects and write short scripts, and should understand how the object hierarchy works. If you have gone through the "Getting Started" tutorial, you have enough information to fully understand this topic.

Contents:

- The Eight Colors of an Object
- Color Inheritance
- Specifying a Color
- System Color Capabilities
- Summary

The Eight Colors of an Object

Each Revolution object has eight color "slots", eight properties of the object that together specify the color of all the object's parts. These properties are:

- `foregroundColor`: the color of text in the object
- `backgroundColor`: the color the object is filled with
- `borderColor`: the color of the object's borders if its `threeD` property is false
- `topColor`: the color of a 3-D object's raised edge
- `bottomColor`: the color of a 3-D object's lowered edge
- `shadowColor`: the color of an object's drop shadow
- `hiliteColor`: the color of highlighted objects and of text selections
- `focusColor`: the color of the outline around the object when it has the focus

You can see and set all eight of these colors at once using the colors property, or see and set each property individually. You can also set the colors of an object in the "Colors & Patterns" pane of the object's property inspector.

Since each object type has eight colors available, some color properties do double duty for some objects. For example, the backgroundColor of a scrolling field determines both the background color under the text and the color of the arrow boxes at the ends of the scroll bar.

(For a detailed description of each color property's effect on each object type, see the entry in the Transcript Dictionary for each color property.)

Pattern properties

Each object also has eight pattern properties corresponding to the eight color properties:

foregroundPattern: the pattern of text in the object (Unix systems)

backgroundPattern: the pattern the object is filled with

borderPattern: the pattern of the object's borders if its threeD property is false

topPattern: the pattern of a 3-D object's raised edge

bottomPattern: the pattern of a 3-D object's lowered edge

shadowPattern: the pattern of an object's drop shadow

hilitePattern: the pattern of highlighted objects and of text selections

focusPattern: the color of the pattern around the object when it has the focus

Each pattern property can be set to the ID of an image. The image is tiled to create a pattern.

Important! If you set a pattern property for an object, the corresponding color property stops having an effect until it is set again. Either the color or the pattern can be seen, but not both at once.

Color Inheritance

The color properties of an object are inherited from the settings of the object above it in the object hierarchy. Grouped controls inherit from their group, controls inherit from the card they're on, cards inherit from the stack they're in, and substacks inherit from their main stack.

For example, if a field's backgroundColor property is not specified, it takes on the background color of the card that owns it. If no backgroundColor is specified for the card either, the stack's backgroundColor is used. This means you can set a backgroundColor for a stack, and every object in it will automatically use that background color.

When an object does not have a setting for a color property, that property reports a value of empty. For example, if a card's shadowColor is empty, the card inherits the stack's shadowColor. If a color property of an object is non-empty, that color overrides any color the object might inherit from an object above it in the object hierarchy.

The effective keyword

If a color property for an object is empty, this indicates that the color is inherited from another object further up the object hierarchy. If you need to find out what color an object displays, but that color property for the object is empty, use the effective keyword to find out the actual color that's displayed in the object.

For example, the following statement traverses the object hierarchy as necessary to find a field's backgroundColor, whether that color is set for the field or for an object that owns the field:

```
get the effective backgroundColor of field "My Field"
```

The hiliteColor property

The hiliteColor property is the single exception to the rules of inheritance for color properties. The hiliteColor is both a global and an object property. If an object's hiliteColor is empty, it uses the global hiliteColor setting, instead of inheriting from the object above it in the object hierarchy.

Specifying a Color

Using Transcript, you can refer to a color by name, as a numeric triplet, or as an HTML-style color reference. You can use any of these methods to refer to a color.

All the ways of specifying a given color are equivalent and may be used interchangeably. For example, the color name "white", the numeric triplet "255,255,255", and the HTML color reference "#FFFFFF" all describe the same color.

Tip: You can use the is a operator to find out whether a certain string is accepted as a color reference by Transcript.:

```
if myColor is a color then put myColor into it
if field "Entry" is not a color then beep
```

Tip: To find out what the color is at a certain point on the screen, enter the following into the message box:

```
the mouseColor
```

Then move the mouse pointer over the color you want to check and press the Return key.

Specifying a color by name

Transcript provides a set of standard color names you can use with any color property. To see these colors, their names, and the equivalent as numeric triplets and HTML color references, see the Color Names Reference (in the See Also menu at the top of this window).

You can get a list of allowed color names by using the colorNames function in a handler or the message box.

Note: Not all the possible colors have names.

Specifying a color as a numeric triplet

Since computer monitors display colors in terms of the additive primary colors (red, green, and blue (RGB)) you can specify a color numerically by giving the amount of each primary that makes up the color. This kind of color specification is called a numeric triplet.

The three numbers in a numeric triplet are separated by commas. Each of the three numbers must be an integer between zero (none of the color) and 255 (maximum intensity of the color).

For example, the triplet "255,0,127" means 255 units of red, zero of green, and 127 of blue. 255 is the maximum intensity, so the color being described has the maximum intensity of red in it, no green at all, and half the possible intensity of blue. This color is a reddish shade of purple.

Shades of gray:

Since the three amounts are primary colors for the RGB system, equal numbers for all three colors produce a shade of gray. For example, `127,127,127` specifies a medium gray shade; `64,64,64` specifies a dark gray; and `196,196,196` specifies a light gray.

If all three numbers are zero, the resulting color is black. If all three numbers are the maximum of 255, the resulting color is white.

Primary and secondary colors:

Pure red is specified when the second and third numbers for green and blue are zero. The higher the first number, the brighter the red. `255,0,0` specifies the brightest possible red; `150,0,0` specifies a darker red. Similarly, you can specify pure green or pure blue by setting the other two numbers in the triplet to zero. You make secondaries by setting only one of the numbers to zero: `255,255,0` is bright yellow, `255,0,255` is bright magenta, and `0,255,255` is bright cyan (a blue-green color).

The ability to tweak the setting of each primary color independently means you can achieve very subtle color effects with numeric triplets. For example, the color becomes brighter as you increase the numbers, and darker as you decrease them. Colors are pure when one or two numbers are zero, and become duller as the numbers get closer together, until the color becomes completely gray when all three numbers are equal.

Specifying a color as an HTML color reference

HTML color references are similar to numeric triplets. Each has three numbers, to specify red, green, and blue. However, in an HTML color specification, the numbers are expressed as hexadecimal (base-16) 2-digit numbers, rather than ordinary decimal numbers, and are not separated by commas. HTML color specifications begin with a hash mark (#).

Note: You can use either uppercase or lowercase letters for the hexadecimal digits A-F. Uppercase is used by convention.

For example, the HTML color reference "#FF007F" means FF (hexadecimal) units of red, 00 of green, and 7F of blue. FF in hexadecimal is equal to 255 in decimal numbers, 00 is zero, and 7F hexadecimal is equal to 127 decimal. This HTML color reference, therefore, specifies the same color as the numeric triplet "255,0,127": a reddish purple.

Shades of gray:

As with numeric triplets, since the three amounts are primary colors for the RGB system, equal numbers for all three colors produce a shade of gray. For example, `#7F7F7F` specifies a medium gray shade; `#404040` specifies a dark gray; and `#C4C4C4` specifies a light gray.

If all three numbers are 00, the resulting color is black. If all three numbers are the maximum of FF, the resulting color is white.

Primary and secondary colors:

Primary and secondary colors work much the same way as for numeric triplets. `#FF0000` specifies the brightest possible red; `#960000` specifies a darker red. Similarly, you can specify pure green or pure blue by setting the other two hexadecimal numbers to 00. You make secondaries by setting only one of the numbers to 00: `#FFFF00` is bright yellow, `#FF00FF` is bright magenta, and `#00FFFF` is bright cyan (a blue-green color).

Important! Although it is good programming practice to enclose all color specifications in double quotes anyway, you must always quote HTML color specifications, because otherwise the leading # is interpreted as a comment delimiter. This will cause Revolution to ignore the rest of the line, and probably cause a script error.

System Color Capabilities

Parts of your application may depend on the color capabilities of the current system—such as how many colors the system can display—and on whether you can change the color table to suit your application.

For example, if your application uses photographic icons that don't look good in black and white, you can detect whether the current system is black-and-white and substitute appropriate icons when your application starts up.

Screen colors

On computer systems, each pixel can be a different color. In general, the total number of possible colors is 2^{24} , approximately 16 million. This number is the sum of all combinations of three numbers from zero to 255. (And now you know why numeric triplet values run from zero to 255 for each of the three primaries red, green, and blue.) The number of these possible colors that can actually be displayed depends on the video system hardware and the current system settings, but is always a power of two.

At one extreme, some video systems can display only two colors: black and white. These are referred to as 1-bit displays, because each pixel's color can be described in a single bit of data: zero for black or 1 for white.

At the other extreme, some video systems can display all of the 16 million possible colors; that is, each pixel can have any of these colors at any time. These are referred to as 24-bit systems, because it takes a total of 24 bits to specify each of the available colors. Most new computer systems have 24-bit displays, although many are initially set up to display fewer colors.

In between, some video systems can display 2^8 colors (256), 16 colors, or 4 colors.

True color and color tables

24-bit and 16-bit video systems are called "true color" or "direct color" systems, because the color of each pixel is specified as a numeric triplet in the set of 16 million possible colors.

Systems that display 4, 16, or 256 colors use a structure called a color table to determine which colors are available, and to specify the color of each pixel. A color table is a subset of the 16 million possible colors.

For example, the color table for a 256-color system contains 256 colors, each of which comes from the set of 16 million possible colors. Each pixel is one of the 256 colors in the color table; the color of each pixel is specified as a number in the color table, instead of a color specification. Similarly, a 16-color system uses a color table with 16 entries, and a 4-color system uses a color table with four entries.

Most systems allow an application to change the colors in the color table. A few provide a fixed color table that can't be changed by individual applications.

Finding out what the system can display

Transcript includes a number of properties and functions you can use to find out the current system's current settings for color display in detail. You can either adjust your application to display properly with the current settings, or ask the user to change them.

Note: The color settings are updated only when you start up the application. If you change the settings after starting up the application, you must quit and restart to update the function and property values.

Black and white, or color?

The `colorWorld` property returns true if the screen is displaying color (or grayscale), and false if it is black-and-white only.

How many colors?

The `screenColors` function returns the number of colors the screen can display. The `screenDepth` function returns the bit depth of the screen. (The two functions are closely related, because the number of colors is equal to 2 raised to the bit depth.)

If the `screenColors` function returns 2, or the `screenDepth` returns 1, the screen is black-and-white only.

Can the color table be changed?

The `screenType` function returns additional information on the kind of color table the system uses. If the `screenType` returns `ôGrayScaleö` or `ôPseudoColorö`, the system uses a color table that can be changed by the application. If the `screenType` returns `ôStaticGrayö` or `ôStaticColorö`, the color table is fixed and cannot be changed.

Changing the color table

Most systems that use color tables allow applications to change the color table on the fly. If the `screenType` function returns `ôGrayScaleö` or `ôPseudoColorö`, you can set the `colorMap` property to the color table you want to use.

The `colorMap` consists of as many color references as there are colors in the color table, one per line. You can set the lines in the color table to any color you want. For example, if your application uses a color scheme based on a sunrise, you can set the color table to one that includes many shades of orange and red.

When Revolution displays an image, player, or video clip on a system that uses a color table, it automatically adjusts the color table, giving priority to the colors in the image or video so that it is shown to best advantage. If you prefer to avoid this behavior and control the color table fully, set the `lockColorMap` property to true.

Summary

In this topic, you have learned that:

ò Each object has eight color properties, which specify all the colors used for that object's various parts. Each object also has eight pattern properties that correspond to the color properties and can be used instead of a solid color.

ò If one of an object's colors is not specified, it inherits that color property from the next object in the object hierarchy.

ò You can specify a color by name (white), as a numeric triplet (0,0,0), or as an HTML-style color spec (#000000). All three methods of specifying a color are equivalent.

ò A system running your application may be capable of displaying anywhere from 16 million colors to black and white only. If your application relies on color, you can check the current system settings, as well as changing the color table.

About object types and object references

See Also:

About messages and the message path, Memory and Limits Reference, How to find out the object type of an object, How to refer to a control on another card, Tools menu > Application Browser, Object menu > Object Inspector, control keyword, ID property, marked keyword, mouseControl function, mouseStack function, name property, number property, owner property, recent keyword, templateAudioClip keyword, templateButton keyword, templateCard keyword, templateEPS keyword, templateField keyword, templateGraphic keyword, templateGroup keyword, templateImage keyword, templatePlayer keyword, templateScrollbar keyword, templateStack keyword, templateVideoClip keyword

Objects are the building blocks of a Revolution application. Everything you see on the screen, from the menu bar to the stack window to the buttons and fields in it, is an object.

This topic discusses each of Revolution's twelve object types, how to refer to objects in a handler, and how to group objects together.

To fully understand this topic, you should know how to create controls and understand how the object hierarchy works. If you have gone through the "Getting Started" tutorial, you have enough information to fully understand this topic.

Contents:

- Revolution Object Types

- Stacks

- Cards

- Buttons

- Fields

- Scrollbars

- Images

- Graphics

- Players

- EPS Objects

- Audio Clips

- Video Clips

- Groups

- Referring to Objects

Revolution Object Types

Revolution has twelve object types: stacks, cards, buttons, fields, scrollbars, images, graphics, players, EPS objects, audio clips, video clips, and groups. Each object type has its own appearance and behavior, its own collection of properties, and its own uses.

Every Revolution object belongs to one of these types.

Stacks

A stack is the basic Revolution object. Each window corresponds to a stack.

Each Revolution stack file contains at least one main stack, and may contain one or more substacks. Main stacks and substacks are identical in structure and appearance, and in the kinds of objects they can hold.

Main stacks contain at least one card, and may contain backgrounds, audio clips, video clips, and substacks. Main stacks are not contained by any other object, and are at the top of the object hierarchy.

Substacks contain at least one card, and may contain backgrounds, audio clips, and video clips. Substacks are contained in a main stack.

Stack references

A stack can be referred to as a stack, window, or wd:

```
show stack "My Stack"
go card 3 of wd "Some Stack"
set the name of window 3 to "Help"
```

A note about stack names:

Stack names starting with `örevö` are reserved by the Revolution development environment.

Cards

A card is a single panel of a stack. One card of a stack can be displayed in the stack's window at any given time.

Cards may contain any combination of buttons, fields, images, graphics, scrollbars, EPS objects, players, and groups. Cards are contained in a stack.

A card can be referred to as a card or cd:

```
go card 3
set the backgroundColor of this cd to "black"
```

Buttons

A button is a clickable object that appears on a card. Depending on the setting of their style and menuMode properties, buttons can be push buttons, radio buttons, checkboxes, or menus.

Buttons do not contain any other objects, and are at the bottom of the object hierarchy. Buttons are contained in a group or card.

Button references

A button can be referred to as a button or btn:

```
select button "Font"
set the top of btn myButton to zero
```

Button menu references

If the button is a button menu that's being displayed in the menu bar, you can use the word `ömenuö` to refer to it:

```
get menuItem 2 of menu "Edit"
```

-- same as 'get line 2 of button "Edit"'

Because menus are also buttons, you can use a button reference to get the same information. But you may need to specify the group and stack the button is in, to avoid ambiguity. (For example, if there is a standard button named `edit` on the current card, the expression `button "Edit"` refers to that button, not to the one in the menu bar.) An unambiguous button reference to a menu might look like this:

get line 2 of button "Edit" of group "Menu" of stack "Main"

The above statement produces the same information as the form using `menuItem`, but you need to know the group name and possibly which stack it's in, so the `menuItem` form is a little more convenient.

Fields

A field is a rectangular object that contains styled text and appears on a card.

Fields do not contain any other objects, and are at the bottom of the object hierarchy. Fields are contained in a group or card.

Field references

A field can be referred to as a field or fld:

- select after text of field "Content"
- set the `showBorder` of fld 3 to false

Scrollbars

A scrollbar is an object with a movable part to indicate the position of the scrollbar. Scrollbars appear on a card and may be displayed vertically or horizontally by changing their height and width properties.

Scrollbars do not contain any other objects, and are at the bottom of the object hierarchy. Scrollbars are contained in a group or card.

Images

An image is an object that contains bitmapped data and appears on a card.

Images do not contain any other objects, and are at the bottom of the object hierarchy. Images are contained in a group or card.

An image can display either its own data, contained in the image object itself, or data contained in an external picture file.

Images can display data in GIF, JPEG, PNG, BMP, XWD, XBM, XPM, PBM, PGM, or PBM formats. On Mac OS and OS X systems, PICT files can also be displayed (but they cannot be displayed on Unix or Windows systems).

To place data in an image, either paint in the image with the Paint palette, or import a picture file (by either using the import command or choosing File menu Import as Control Image File).

To display an external picture file in an image, either set the image's `fileName` property to the file path of the file, or choose File menu New Referenced Control Image File.

Image references

An image can be referred to as an image or img:

```
get the rect of image "Splash"  
put myPictureData into img ID 2231
```

Graphics

A graphic is a resizable vector shape that appears on a card. Depending on the setting of their style property, graphics can be rectangles (or squares), ovals (or circles), lines, regular polygons, irregular polygons, or freehand curves.

Graphics do not contain any other objects, and are at the bottom of the object hierarchy. Graphics are contained in a group or card.

A graphic can be referred to as a graphic or grc:

```
select graphic "Divider Line"  
set the visible of grc thisGrc to false
```

Players

A player is an object that displays an external QuickTime file and appears on a card.

Players do not contain any other objects, and are at the bottom of the object hierarchy. Players are contained in a group or card.

A player has no data of its own. The movie or sound displayed by a player comes from an external file. To specify the file to be displayed by a player, either set the playerÆs fileName property to the file path of the file, or choose File menu New Referenced Control QuickTime-Supported File.

A player is referred to as a player:

```
set the rect of player "Bouncing Betty" to myRect
```

EPS Objects

An EPS object is a rectangular object that contains Encapsulated PostScript data and appears on a card. EPS objects are supported only on Unix systems that support Display PostScript.

EPS objects do not contain any other objects, and are at the bottom of the object hierarchy. EPS objects are contained in a group or card.

An EPS object is referred to as an EPS:

```
delete EPS "Form Layout"
```

Audio Clips

An audio clip is an object that contains audio data. Audio clips have no visual representation and do not appear on a card.

Audio clips do not contain any other objects, and are at the bottom of the object hierarchy. Audio clips are contained in a stack.

An audio clip can be referred to as an audioClip or ac:
play audioClip "Eek!"
set the playLoudness of audioClip ID 2234 to 10

Video Clips

A video clip is an object that contains video data. Video clips have no visual representation and do not appear on a card.

Video clips do not contain any other objects, and are at the bottom of the object hierarchy. Video clips are contained in a stack.

A video clip is referred to as a videoClip or vc:
play videoClip "Munch Animated Scream"
set the script of vc 10 to field "Last Clip Script"

Groups

A group is an object that contains other controls and appears on one or more cards.

Groups may contain any combination of buttons, fields, images, graphics, scrollbars, EPS objects, players, and other groups. Groups can be contained in other groups or in a card.

A group can be referred to as a group or a grp:

set the vScroll of group "Main" to 50
select group ID 844

A group can also be referred to as a background, bkgnd, or bg. When using one of these terms to refer to a group, the reference is to one among the groups in a stack, rather than to one among the groups on a card. This means that you can refer to a group of a card, but not a group of a stack:

group 3 of this card -- valid reference
group "Bar" of card 3 of stack "Graphics" -- valid reference
group "Bar" of stack "Graphics" -- NOT VALID

Conversely, you can refer to a background of a stack, but not a background of a card:

background 3 of this card -- NOT VALID
background "Bar" of card 3 of stack "Graphics" -- NOT VALID
background "Bar" of stack "Graphics" -- valid reference

For more information about group objects, see the topic [About groups and backgrounds](#).

Referring to Objects

In general, you can refer to any object by its name, number, or ID property.

Referring to objects by name

You can refer to an object using its object type followed by its name. For example, to refer to a button named "OK", use the phrase button "OK":

set the loc of button "OK" to 32,104

To change an object's name, enter a name in the object's property inspector, or use the set command to change the object's name property:

set the name of field "Old Name" to "New Name"
select after text of field "New Name"

Referring to objects by number

A control's number is its layer on the card, from back to front. A card's number is its position in the stack. A stack's number is the order of its creation in the stack file. (A main stack's number is always zero.)

You can refer to an object using its object type followed by its number. For example, to refer to the third-from-the-back field on a card, use the phrase field 3:

set the backgroundColor of field 3 to blue

To change the number of a card or control, change the Layer box in the Size & Position pane of object's property inspector, or use the set command to change the object's layer property:

set the layer of field "Backmost" to 1

Tip: New objects are always created at the top layer. To refer to an object you've just created, use the ordinal last:

create button
set the name of last button to "My New Button"

Referring to objects by ID

Each object in Revolution has an ID number. The ID property never changes (except for stack IDs), and is guaranteed unique within the stack: no two objects in the same stack can have the same ID property.

You can refer to an object using its object type followed by its ID number. For example, to refer to a card whose ID property is 1154, use the phrase card 1154:

go to card ID 1154

You cannot change an object's ID property (except for a stack).

Referring to objects by ordinal

You can refer to an object using its object type followed by the ordinal numbers first through tenth, or the special ordinals middle and last. To refer to a random object, use the special ordinal any. For example, to refer to the last card in the current stack, use the special ordinal last:

go to last card

The special descriptor this

Use the this keyword to indicate the current stack, or the current card of a stack:

```
set the backgroundColor of this stack to white
send "mouseUp" to this card
set the textFont of this card of stack "Menubar" to "Sans"
```

Control references

A control is any object that can appear on a card. Fields, buttons, scrollbars, images, graphics, players, EPS objects, and groups are all controls. (Stacks, cards, audio clips, and video clips are not controls.)

You can refer to an object of any of these object types using the word `control`, followed by an ID, name, or number:

```
hide control ID 2566
send mouseDown to control "My Button"
set the hilite of control 20 to false
```

If you use a name, as in the expression `control "Thing"`, the reference is to the first control (with the lowest layer) that has that name.

When you refer to a control by number using its object type, the reference is to the Nth control of that type. For example, the phrase `field 1` refers to the lowest field on the card. (This may not be the lowest control, because there may be controls of other types underneath field 1.) However, when you refer to a control by number using the word `control`, the reference is to the Nth control of any type. The phrase `control 1` refers to the lowest control on the card, which may be of any type.

Tip: To refer to the object underneath the mouse pointer, use the `mouseControl` function.

Nested Object References

To refer to an object that belongs to another object, nest the references in the same order as the object hierarchy. For example, if there is a button called `My Button` on a card called `My Card`, you can refer to the button like this:

```
show button "My Button" of card "My Card"
```

You can mix names, numbers, ordinal references, and IDs in a nested object reference, and you can nest references to whatever depth is required to specify the object. The only requirement is that the order of references be the same as the order of the object hierarchy, going from an object to the object that owns it. Here are some examples:

```
field ID 34 of card "Holder"
player 2 of group "Main" of card ID 20 of stack "Demo"
first card of this stack
stack "Dialog" of stack "Main" -- "Dialog" is a substack
```

If you don't specify a card in referring to an object that is contained by a card, Revolution assumes the object is on the current card. If you don't specify a stack, Revolution assumes the object is in the current stack. You can reference a control in another stack by either of the following methods:

ò Use a nested reference that includes the name of the stack:

field 1 of stack "My Stack"
graphic "Outline" of card "Tools" of stack "Some Stack"

ò Set the defaultStack property to the stack you want to refer to first. The defaultStack specifies the current stack, so you can refer to any object in the defaultStack without including a stack name. This example sets a checkbox in the current stack to have the same setting as a checkbox in another stack called "Other Stack":

```
put the defaultStack into savedDefault
-- so you can set it back later
set the defaultStack to "Other Stack"
put the hilite of button "Me" into meSetting
-- this button is in "Other Stack"
set the defaultStack to savedDefault
set the hilite of button "Me Too" to meSetting
-- this button is in the original stack
```

If an object is in a group, you can include or omit a reference to the group in a nested reference to the object. For example, suppose the current card contains a button called "Guido", which is part of a group called "Stereotypes". You can refer to the button with any of the following expressions:

```
button "Guido"
button "Guido" of card 5
button "Guido" of group "Stereotypes"
button "Guido" of group "Stereotypes" of card 5
```

If there is no other button named "Guido" on the card, these examples are equivalent. If there is another button with the same name in another group (or on the card, but not in any group), you must either specify the group (as in the second and third examples) or refer to the button by its ID property, to be sure you're referring to the correct button.

About main stacks, substacks, and the organization of a stack file

See Also:

About windows, palettes, and dialog boxes, How to move a stack to another file, Why is there already a stack with the same name?, File menu > New Mainstack, File menu > New Substack of (main stack name), File menu > New Referenced Control, File menu > Move Substack to File..., Tools menu > Application Browser, create stack command, mainStack property, openStacks function, revLoadedStacks function, stacks function, substacks property

The stack is the basic unit of organization of a Revolution application. Each application has at least one stack, and may have many more—either in the same file, or in other stack files.

This topic discusses the structure of a stack file, the relationship between stacks that reside in the same stack file, and how to refer to and use files outside your application's stack file.

To fully understand this topic, you should know how to open, create, and save a stack. If you have gone through the "Getting Started" tutorial, you have enough information to fully understand this topic.

Contents:

- The Structure of a Stack File

- Main Stacks and Substacks

- Stacks, Stack Files, and Memory

- Using External Files

The Structure of a Stack File

Each Revolution file contains one or more stacks: either a single main stack, or a main stack and one or more substacks. Since each stack is a window (including editable windows, modeless and modal dialog boxes, and palettes), a single stack file can contain multiple windows.

You can use this capability to bundle several related stacks into a single file for easy distribution, to organize your stacks into categories, or to allow several stacks to inherit properties from the same main stack.

Opening a stack file

When you open a stack file, either by using the "Open Stack" menu item in the File menu or by using the go, modal, modeless, palette, or topLevel commands, the stack file's main stack opens automatically to its first card.

Substacks in the stack file do not open automatically when the stack file opens. You must open a substack in a handler or the message box, or by Control-clicking (Mac OS or OS X) or right-clicking (Unix or Windows) the substack's name in the Application Browser window and choosing "Go" from the contextual menu.

Saving a stack file

A stack file is saved as a whole. If you save a stack, all the other stacks in the same stack file are saved at the same time.

Main Stacks and Substacks

The first stack created in a stack file is called the main stack. Any other stacks created in the same stack file are called substacks of the main stack.

The main stack is part of the object hierarchy of all other stacks in the same stack file. In other words, the main stack contains its substacks. Messages that are not handled by a substack are passed on to the main stack's script, and color and font properties are inherited from the main stack by its substacks.

Dialog boxes and palettes are commonly stored as substacks of the main application window, which is typically a main stack. This allows you to store handlers used by all the substacks in the main stack's script. Because main stacks are part of the object hierarchy of their substacks, the substacks can call these handlers from scripts within the substack.

Stacks, Stack Files, and Memory

A stack file can be loaded into memory without actually being open. A stack whose window is closed (not just hidden) is not listed in the `openStacks` function. However, it takes up memory, and its objects are accessible to other stacks. (For example, if a closed stack that's loaded into memory contains a certain image, you can use the image as a button icon in another stack.)

Note: If one stack in a stack file is loaded into memory, so are any other stacks in the same stack file. You cannot load one stack in a stack file without loading all the rest at the same time, even if you open only one of the stacks.

A stack can be loaded into memory without being open under the following conditions:

- ò A handler in another stack referred to a property of the closed stack. This automatically loads the referenced stack into memory.
- ò The stack is in the same stack file as another stack that is open.
- ò The stack was opened and then closed, and its `destroyStack` property is set to false. (If the `destroyStack` property is false, the stack is closed, but not unloaded, when its window is closed.)

Tip: To list all stacks in memory, whether they're open or closed, use the `revLoadedStacks` function.

Using External Files

Since each loaded stack file takes up as much memory as the size of all its stacks, it is often advisable to place large, seldom-used objects (such as color pictures, sound, and video) in external files, which can be bundled with your application and loaded into memory only when you need them.

There are two ways to do this:

- ò Keep media such as images, sounds, and video in external files, in the appropriate format, and use referenced controls to display the contents of the files. When you use a referenced control, the image, sound, or video file is loaded into memory only when a card that contains the referenced control is being displayed. The needed memory for the image, sound, or video is therefore only used when actually needed.

To create a referenced control, use the "New Referenced Control" submenu in the File menu, or create an empty image or player object, then set the object's fileName property to a file path for the file you want to reference.

• Keep portions of your application in a separate stack file, and refer to the stacks in that stack file as necessary. The stackFiles property simplifies referring to external stack files. When you set a stack's stackFiles property to include one or more file paths, the stacks at those locations become available to your stack by simply referring to the stacks by name.

About groups and backgrounds

See Also:

About messages and the message path, How to automatically include groups on a new card, How to display objects on more than one card, How to include a group on a card, Why isn't a group listed?, Tools menu > Application Browser, Object menu > Group Selected, Object menu > Remove Group, Object menu > Place Group, backgroundBehavior property, group command, groupIDs property, groupNames property, place command, remove command, selectGroupedControls property

Groups, Revolution's most versatile object type, are used for several purposes: radio button clusters, menu bars, and sets of objects that are shared between cards.

This topic discusses how to create and work with groups, how to refer to them in handlers, and how to add or delete a shared group from a card.

To fully understand this topic, you should know how to create new controls and understand how the message path works. If you have gone through the "Getting Started" tutorial, you have enough information to fully understand this topic.

Contents:

What Is a Group?

Groups and Backgrounds

Nested Groups

Selecting and Editing Groups

Adding and Deleting Groups

Groups and the Message Path

What Is a Group?

A group is a single object that holds a set of objects. You group objects by selecting the controls you want to include in the group, then using the group command or choosing Object menu Group Selected.

Once you've created the group, it becomes an object in its own right. You can select, copy, move, and resize the group, and all the objects in the group come with it. The objects in the group maintain their own identities, and you can add objects to the group or delete them, but the objects are owned by the group instead of the card.

A group has its own properties and its own script. Groups can be any size, can be shown or hidden, and can be moved to any location in the stack window, just like any other control. Like other controls, groups can be layered in any order with the other controls on the card.

Unlike other controls, however, groups can appear on more than one card. You place a group on a card using the place command or the "Place Group" submenu in the Object menu. A group that's shared between cards appears at the same place on each of them. A change made to a shared group while you're on one card is reflected on all the other cards that share the group.

Groups and Backgrounds

Both the term `group` and the term `background` can be used to refer to groups. The terms are usually interchangeable, but not always.

In general, the term `group` refers to groups that are placed on a card, while the term `background` refers to all the groups in a stack. The expression `the number of groups` evaluates to the number of groups on the current card. The expression `the number of backgrounds` evaluates to the number of groups in the current stack, including groups that are not placed on the current card.

When you refer to a group by number, if you use the word `group`, the number is interpreted as referring to the groups on the referenced card, in order by layer. If you use the word `background`, the number is interpreted as referring to the groups in the stack, in the order of their creation.

For example, the expression `the name of group 1` evaluates to the name of the lowest-layered group on the current card, while the expression `the name of background 1` evaluates to the name of the first group that was created in the stack—whether or not that particular group is placed on the current card, or appears on any card at all.

However, when you use the word `background` in an object reference, it is synonymous with the word `group`.

Finally, the term `background` can be used to refer to the set of cards that share a particular group. The following statement goes to the third card on which the group named `Navigation` is placed:

```
go card 3 of background "Navigation"
```

Nested Groups

Revolution supports nested groups (one group containing another). Since a group is itself a control, it can be contained in another group.

Creating a nested group is just like creating a group: select the controls you want to group (including the existing group), then choose Object menu Group Selected. The existing group is now a member of the new group.

Selecting and Editing Groups

When you click a grouped control, you can select either the control itself or the group of which it's a part. Which one is selected is controlled by the `selectGroupedControls` property. If this property is true, clicking a grouped control selects only the control. If it is false, clicking any control in a group selects the group.

In other words, if you want to work with individual controls without regard to whether they're part of a group, set the `selectGroupedControls` to true. If you want to select and work with a group as an object, set the `selectGroupedControls` to false.

You can change the `selectGroupedControls` setting by choosing Edit menu Select Grouped Controls. You can also reverse the current setting momentarily by holding down the Command key (on Mac OS or OS X systems) or Control key (on Unix or Windows systems) while clicking.

Tip: If a group's `showBorder` property is set to true, an outline appears at the group's edges. However, clicking within or on the border does not select the group. To select the group, you must click one of its controls.

To change a group's properties, select the group and choose Object menu Object Inspector.

To change the arrangement of controls in a group, use the start editing command, or select the group and choose Object menu Edit Group. The other controls on the card disappear to make it easier for you to edit the group. In group-editing mode, you can select individual controls in the group regardless of the setting of the `selectGroupedControls` property. Any new controls you create in this mode will be added to the group being edited. When you are finished editing the group, choose Object menu Stop Editing Group.

Adding and Deleting Groups

Once you create a group, you can display it on any or all cards in the stack. To display a group on a card, either use the place command in a handler or the message box, or choose the group's name from the `Place Group` submenu in the Object menu. (The `Place Group` submenu displays only groups that are not already on the current card. You cannot place a group on a card twice.)

Tip: When you create a new card, if there are any groups on the current card whose `backgroundBehavior` property is true, they are automatically placed on the new card. To make it easy for all the cards in a stack to share a single group, create the group on the first card and set its `backgroundBehavior` to true, before you create any other cards.

To remove a group from the current card without deleting it from the stack, use the remove command, or select the group and choose Object menu Remove Group. The group disappears from the current card, but it's still placed on any other cards that share the group.

Removing a group from every card in the stack does not remove it from the stack: it is still part of the stack and can still be placed on cards. You can also use the start editing command to edit the group. (Since the group is not on any card, you must refer to it using the term `background` instead of the term `group`.)

To delete a group completely from the stack, use the delete command, or select the group and choose Edit menu Clear. Deleting the group removes it from all the cards it appears on, and from the stack itself.

To dissolve a group back into its component controls, select the group and either use the ungroup command or choose Object menu Ungroup. Ungrouping deletes the group object and its properties (including its script) from the stack, but does not delete the controls in it. Instead, they become card controls of the current card. The controls disappear from all other cards the group is on.

Tip: If you ungroup a group, then select the controls and regroup them before leaving the current card, the group is restored as it was. However, leaving the current card makes the ungrouping permanent and deletes the group from all other cards it was on.

Groups and the Message Path

A group's position in the message path depends on whether its `backgroundBehavior` property is set to true or false.

If a group's `backgroundBehavior` is false, the group is in the message path for all controls it owns, but is not in the message path of any other object.

If a group's `backgroundBehavior` property is true, the group is also in the message path for any cards it is placed on. If you send a message to a card control, the message passes through the control, then the card, then any groups on the card whose `backgroundBehavior` is true (in order by group number), then the stack. (But if a group has already received a message because it was originally sent to one of the controls in the group, the message is not sent through the group again after the card has handled it.)

Since a group owns any controls that are part of the group, the group is in the message path for its own controls, regardless of whether its `backgroundBehavior` is true or false.

Tip: If you want a handler in a group's script to affect only the objects in the group, place the following statement at the beginning of the handler:

```
if the owner of the target is not me then pass message
```

This filters out any objects that are not part of the group.

About windows, palettes, and dialog boxes

See Also:

About main stacks, substacks, and the organization of a stack file, About menus and the menu bar, How to change a window's title, How to change the size and position of a window, How to create a custom dialog box, How to display a dialog box, How to edit a palette or dialog box, How to make a throbbing button, How to make the Return or Enter key activate a button, How to show or hide a window, How to remove a window's close box, How to respond to closing a window, How to respond to moving a window, How to respond to resizing a window, Why do menu commands affect the wrong stack?, Why is the card number in the window name?, Why is there an asterisk in the window name?, Window menu > Send Window to Back, Object menu > Stack Inspector, Shortcut to display a contextual menu, answer command, ask command, decorations property, destroyWindow property, label property, modal command, mode property, modeless command, palette command, sheet command, topLevel command, style property

In Revolution, each window is a stack. This includes editable windows, modeless and modal dialog boxes, and palettes, as well as subwindows available on some operating systems, such as sheets and drawers.

This topic discusses the way windows are implemented in Revolution applications; how to change window appearance; and how to control the behavior of the various window types.

To fully understand this topic, you should know how to create stacks and change properties. If you have gone through the "Getting Started" tutorial, you have enough information to fully understand this topic.

Contents:

- Window Types and the Mode of a Stack
- Stack Decorations and Window Appearance
- Window Placement and Layering
- Open, Closed, and Hidden Windows
- Summary

Window Types and the Mode of a Stack

The nature of a stack's window depends on the stack's style property and on what command was used to open the stack. You can either specify the window type when opening the stack, or allow the stack's style property to determine the type of window it is displayed in.

The standard window types

Revolution windows are usually one of four types: editable windows, modal or modeless dialog boxes, or palettes.

Most windows are editable windows, and this is the default mode for Revolution stacks. If you open a stack using the go command (without specifying a mode), or using the Open Stack menu item, then the stack is displayed as an editable window unless its style property specifies another window type.

Editable windows:

An editable window has the appearance and behavior of a standard document window. It can be interleaved with other windows, and you can use any of Revolution's tools to create, select, move, or delete objects in the stack.

Note: If the stack's `canModify` property is set to true, the window retains its standard appearance, but tools other than the Browse tool can no longer be used in it. (That is, every tool behaves like the Browse tool when clicking in an unmodifiable stack's window.)

To display a stack in an editable window, you use the `topLevel` or `go` commands:

```
topLevel stack "My Stack"
go stack "My Stack" -- "topLevel" is the default mode
go stack "My Stack" as topLevel
```

(Stacks whose style property is set to `ôtopLevelö` always open as editable windows, regardless of what command you use to open them.)

Modeless dialog boxes:

Modeless dialog boxes are similar to editable windows. Like editable windows, they can be interleaved with other windows in the application. Their appearance may differ slightly from the appearance of editable windows, depending on the platform.

Like unmodifiable stacks, modeless dialog boxes do not allow use of tools other than the Browse tool. Use modeless dialog boxes in your application for windows such as a Preferences or Find dialog box, that do not require the user to dismiss them before doing another task.

To display a stack in a modeless dialog box, you use the `modeless` or `go` commands:

```
modeless stack "My Stack"
go stack "My Stack" as modeless
```

(Stacks whose style property is set to `ômodelessö` always open as modeless dialog boxes, regardless of what command you use to open them.)

Modal dialog boxes:

A modal dialog box is a window that blocks other actions while the window is displayed. You cannot bring another window in the application to the front until the dialog box is closed, nor can you edit the stack using the tools in the Tools palette. While a modal dialog box is being displayed, the handler that displayed it pauses until the dialog box is closed.

Modal dialog boxes do not have a close box. Usually, they contain one or more buttons that close the window when clicked.

Tip: If you forgetfully open a modal dialog box without having included a button to close the window when clicked, use the contextual-menu shortcut (Command-Control-Shift-click for Mac OS or OS X, Control-Shift-Right-click for Unix or Windows). This shortcut displays a contextual menu containing the `ôStack Modeö` submenu. This submenu changes the mode of the stack you clicked in, allowing you to switch the stack to another mode without closing it.

To display a stack in a modal dialog box, you use the modal command or go commands:

```
modal stack "My Stack"  
go stack "My Stack" as modal
```

(Stacks whose style property is set to `modal` always open as modal dialog boxes, regardless of what command you use to open them.)

Palettes:

A palette has a slightly different appearance, with a narrower title bar than an editable window. Like dialog box windows, a palette does not allow use of tools other than the Browse tool.

Palettes behave differently than other windows in two ways:

ò A palette floats in front of any editable windows or modeless dialog boxes that are in the same application. Even when you bring another window to the front, it does not cover any palettes. (You can turn this behavior off by setting the `raisePalettes` property to false.)

ò Palette windows disappear when their application is in the background and another application is in front. (You can turn this behavior off by setting the `hidePalettes` property to false.)

Cross-platform note: On some Unix systems, palettes have the same appearance and behavior as ordinary windows and do not float.

To display a stack in a palette, you use the palette command or go commands:

```
palette stack "My Stack"  
go stack "My Stack" as palette
```

(Stacks whose style property is set to `palette` always open as palettes, regardless of what command you use to open them.)

System palettes:

A system palette is like a palette, except that it floats in front of all windows on the screen, not just the windows in its application. Use system palettes for utilities that you want the user to be able to see and access in every application.

To display a stack in a system palette, you set its `systemWindow` property to true:

```
set the systemWindow of stack "My Stack" to true
```

The `systemWindow` setting overrides the stack's style property setting, as well as any window style that was set when you opened the stack.

Note: The system palette style is available only on Mac OS, OS X, and Windows systems.

Subsidiary windows

A subsidiary window is a window that is attached to, or part of, another window. On OS X systems, you can use two kinds of subsidiary windows to display a stack: sheets and drawers.

Sheet dialog boxes:

A sheet is like a modal dialog box, except that it is associated with a single window, rather than the entire application. A sheet appears within its parent window, sliding from underneath the title bar. While the sheet is displayed, it blocks other actions in its parent window, but not in other windows in the application.

To display a stack in a sheet dialog box, you use the sheet command:

```
sheet "My Stack" -- appears in defaultStack
sheet "My Stack" in stack "My Document"
```

The answer, answer file, answer folder, ask, ask file, and answer folder commands all include an ...as sheet form, so you can display these dialog boxes as sheets on OS X.

Cross-platform note: On systems other than OS X, the sheet command displays the stack as an ordinary modal dialog box.

Drawers:

A drawer is a subwindow that slides out from underneath one edge of a window, and slides back in when you close it. You usually use a button in the main window to open and close a drawer.

To display a stack as a drawer, you use the drawer command:

```
drawer "My Stack" at left -- of defaultStack
drawer "My Stack" at bottom of stack "My Document"
```

Cross-platform note: On systems other than OS X, the drawer command displays the stack as an editable window.

Use drawers to hold settings, lists of favorites, and similar controls that are accessed frequently but that don't need to be constantly visible.

Stack menus:

Usually, a menu in a Revolution application is implemented as a button: the text property of the button is used to specify the menu items in the menu, and the button's menuMode property specifies what kind of menu is displayed when the user clicks the button.

It is also possible to display a stack as a pulldown, popup, or option menu. Stack menus are used when a menu needs to contain something other than text. For example, a popup menu containing a slider, or an option menu consisting of icons instead of text, can be implemented as a stack menu.

To display the stack menu, you create a button and set its menuName property to the stack's name. When the user clicks the button, the stack is displayed with the behavior of a menu. Internally, the menu is implemented as a window, and you can use the popup, pulldown, or option command to display any stack as one of these menu types.

(See the topic [About menus and the menu bar](#) for more information about stack menus.)

Window types and the mode property

In a script, you can find out a stack window's type by checking the stack's mode property. This read-only property reports a number that depends on the window type. For example, the mode of an editable window is 1, and the mode of a palette is 4.

You can use the mode property in a script to check what sort of window a stack is being displayed in:

```
if the mode of this stack is 5 then -- modal dialog box
  close this stack
else -- some other type of window
  beep
end if
```

For complete information about the possible values of the mode property, see its entry in the Transcript dictionary.

Changing a window's type

When you use the create stack command, or choose File menu>New Mainstack or File menu>New Substack of (main stack name), the new stack is created as an editable window.

You can use various commands to display a stack in any of the possible window types. You can also change a stack back to an editable window so that you can use the Pointer tool and other tools to make changes to the stack, or to add and delete controls.

Using the window type commands:

Transcript includes several commands that open a stack in a particular type of window:

- topLevel command: opens a stack as an editable window
- modal command: opens a stack as a modal dialog box
- modeless command: opens a stack as a modeless dialog box
- palette command: opens a stack as a palette
- drawer command: opens a stack as a drawer
- sheet command: opens a stack as sheet

You can use any of these commands to open a stack you specify. (If the stack is already open, these commands close it and immediately re-open it in the specified window type.)

There are also several commands to display a stack menu. Usually, you use these commands in a mouseDown handler:

- popup command: opens a stack as a popup menu
- pulldown command: opens a stack as a pulldown menu
- option command: opens a stack as an option menu

Note: If you set a button's menuName property to the name of a stack, the stack menu is displayed automatically when the user clicks the button. You need the popup, pulldown, and option commands only if you want to display a stack menu in some way other than when a button is clicked.

Setting the style property of a stack:

You can also change a window to an editable window, modeless dialog box, modal dialog box, or palette by setting the stack's style property to the desired window type:

```
set the style of this stack to "modeless"
```

Note: If you use any of the commands described above to open a stack, the command temporarily overrides the stack's style property. For example, if a stack's style is `palette` and you open it using the modeless command, the stack opens in a modeless dialog box.

Using the go command:

The most common way of opening a stack is by using the go command:

```
go stack "Hello World"
```

The statement above opens the stack in whatever kind of window is specified by the stack's style property.

You can also use the go command to open a stack as an editable window, modeless or modal dialog box, palette, or sheet by specifying the mode:

```
go stack "Hello World" as palette
go this stack as modal
```

As with the commands discussed above, if you use the go command to open a stack while specifying a mode different from the one indicated by its style, the command temporarily overrides the style property. If you don't specify a mode, the go command opens the stack in the mode specified by its style.

Changing a window's type in the development environment:

You can, of course, use any of the commands described above to change a window's type, either in a handler or in the message box.

Tip: If you're not sure what a stack's name is, you can use the mouseStack function to find out. Enter the following in the message box, then move the mouse pointer over the stack window and press Return:

```
put the mouseStack
```

You can also use the contextual-menu shortcut (Command-Control-Shift-click for Mac OS or OS X, Control-Shift-Right-click for Unix or Windows). This shortcut displays a contextual menu containing the `Stack Mode` submenu. You can use this menu to switch the mode of the stack you clicked to `TopLevel` (editable), `Modeless`, `Modal`, or `Palette`.

Operating-system windows

Some commands (such as ask file and answer file) display a window that is drawn by the operating system. Such standard OS windows are not stacks, and cannot be controlled using the usual window commands and properties.

Cross-platform note: On Unix systems (and on other platforms, if the `systemFileSelector` property is set to true), the `ask file`, `answer file`, and `answer folder` commands display a built-in stack instead of an OS window.

Stack Decorations and Window Appearance

Apart from the differences in appearance between different window modes, the appearance and functionality of a window can vary depending on the stack's properties.

The title bar and its contents

A window's title bar displays the window's title, as well as a close box, minimize box or collapse box, and maximize box or zoom box.

Title bar:

The user drags the title bar of a window to move the window around the screen. In general, if the title bar is not displayed, the user cannot move the window. You use the `decorations` property (discussed below) to hide and show a window's title bar.

When the user drags the window, Revolution sends a `moveStack` message to the current card.

The `decorations` property affects only whether the window can be moved by dragging it. Even if a stack's `decorations` property does not include the title bar decoration, you can still set a stack's `location`, `rectangle`, and related properties to move or resize the window.

Window title:

The title that appears in the title bar of a window is determined by the stack's `label` property. If you change a stack's label in a script, the window's title is immediately updated.

If the label is empty, the title bar displays the stack's `name` property. (If the stack is in an editable window whose `cantModify` is false, an asterisk appears after the window title to indicate this, and if the stack has more than one card, the card number also appears in the window title. These indicators do not appear if the stack has a label.)

Because the window title is determined by the stack's `label` property instead of its `name` property, you have a great deal of flexibility in changing window title. Your scripts refer to the stack by its name—which doesn't need to change—not its label, so you can change the window title without changing any scripts that refer to the stack.

The close box:

The close box allows the user to close the window by clicking it. To hide or show the close box, you set the stack's `closeBox` property:

```
set the closeBox of stack "Bravo" to false
```

When the user clicks the close box, Revolution sends a `closeStackRequest` message, followed by a `closeStack` message, to the current card.

The `closeBox` property affects only whether the window can be closed by clicking. Even if a stack's `closeBox` property is false, you can still use the `close` command in a handler or the message box to close the window.

The minimize box or collapse box:

The terminology and behavior of this part of the title bar varies depending on platform. On Mac OS systems, the collapse box shrinks the window so only its title bar is shown. The minimize box (OS X and Windows systems) or iconify box (Unix systems) shrinks the window to a desktop icon.

To hide or show the minimize box or collapse box, you set the stack's minimizeBox property:

set the minimizeBox of this stack to true

Tip: On OS X and Unix systems, you can set a stack's icon property to specify the icon that appears when the stack is minimized.

When the user clicks the minimize box or collapse box, Revolution sends an iconifyStack message to the current card.

The maximize box or zoom box:

The terminology and behavior of this part of the title bar varies depending on platform. On Mac OS and OS X systems, the zoom box switches the window between its current size and maximum size. The maximize box (Unix and Windows systems) expands the window to its maximum size.

To hide or show the zoom box or maximize box, you set the stack's zoomBox property:

set the zoomBox of stack "Hello" to false

When the user clicks the zoom box or maximize box, Revolution sends a resizeStack message to the current card.

Making a stack resizable

A stack's resizable property determines whether the user can change its size by dragging a corner or edge (depending on operating system) of the stack window.

Some stack modes cannot be resized, regardless of the setting of the stack's resizable property. Modal dialog boxes, sheets, and drawers cannot be resized by the user, and do not display a resize box.

The resizable property affects only whether the window can be resized by dragging a corner or edge. Even if a stack's resizable property is set to false, you can still set a stack's location, rectangle, and related properties to move or resize the window.

When the user resizes a stack, Revolution sends a resizeStack message to the current card.

Tip: To move and resize controls automatically to fit when a stack is resized, use the "Geometry" pane in the control's property inspector.

Window appearance

Details of a window's appearance, such as the width of its title bar and the background color or pattern in the window itself, are mainly determined by the stack's mode. There are a few additional elements of window appearance that you can control with specific properties.

The metal property:

On OS X systems, you use a stack's metal property to give the stack window a textured metal appearance. This metal appearance applies to the stack's title bar and its background.

Note: The metal appearance, in general, should be used only for the main window of an application, and only for windows that represent a physical media device such as a CD player. See Apple's Aqua user-interface guidelines for more information.

The shadow property:

A stack's shadow property controls whether the stack window has a drop shadow. On OS X systems, you can set this property to false to create a window with no shadow.

The windowShape property:

You can use the windowShape property to create a stack window with any shape—including shapes with holes.

To change a window's shape, first create an image (in GIF or PNG format) with a one-bit transparency mask in the shape you want, and import the image into any open stack. When you set a stack's windowShape property to the ID of the image, the stack window's shape changes to the shape of the image's mask.

Window background color:

The background color or pattern of a window's content area—the part that isn't part of the title bar—is determined by the window type and operating system, by default. (For example, on OS X systems, a striped background appears in palettes and modeless dialog boxes.)

If you want a window to have a specific color or pattern, you can set the stack's backgroundColor or backgroundPattern property:

```
set the backgroundColor of stack "Alpha" to "aliceblue"  
set the backgroundPattern of stack "Beta" to 2452 -- img ID
```

This color or pattern overrides the usual color or background pattern.

The decorations property

Most of the properties that pertain to a window's appearance can also be set in the stack's decorations property. The decorations of a stack consists of a comma-separated list of decorations:

```
set the decorations of stack "Gamma" to "title,minimize"
```

The statement above sets the stack's minimizeBox property to true, as well as showing its title bar, and sets other stack properties (maximizeBox, closeBox, metal) to false. Conversely, if you set a stack's minimizeBox property to true, its decorations property is changed to include "minimize" as one of its items. In this way, the decorations property of a stack interacts with its closeBox, minimizeBox, zoomBox, metal, shadow, and systemWindow properties.

The decorations property and menu bars in a window:

On Unix and Windows systems, the menu bar appears at the top of the window. On these systems, whether a window displays its menu bar is determined by whether the `stack's decorations` property includes `menu`:

```
set the decorations of this stack to "title,menu"
```

(On Mac OS and OS X systems, the menu bar appears at the top of the screen, outside any window. On these systems, the `menu` decoration has no effect.)

Decorations and a stack's mode:

The decorations property is independent of the stack's mode (although of course some combinations of decorations are more or less appropriate with different window types). For example, if a stack's decorations property is set to empty, it is displayed without a title bar, whether the stack is displayed as an editable window, modal dialog box, modeless dialog box, or palette.

While the stack's mode is separate from its decorations, the mode may affect whether these properties have an effect. If the decorations property is set to `default`, it displays the appropriate decorations for the current window type on the current platform.

Using window definitions:

Mac OS and OS X systems include special resources, called WDEFs (window definitions), for displaying windows of various kinds. (The exact WDEFs available vary, depending on the operating system version.) You can use the decorations property to access these WDEFs.

To display these special window types, you set the stack's decorations property to a WDEF identifier. A WDEF identifier is the number of a Mac OS WDEF resource multiplied by 16, plus a variation code between zero and 15.

For example, if you have a WDEF resource whose ID is 124, and you want to use its 14th variation (which is number 13, since the numbers start with zero), set the stack's decorations property to 1997. This number is 124 (the resource ID) times 16, plus 13 (the variation code).

Window Placement and Layering

In addition to the normal methods for users to move and re-layer windows, you can use Transcript commands to change a window's position and its front-to-back order among other windows.

Moving a window

Usually, you use either the `location` or `rectangle` property of a stack to move the stack window.

The `location` property specifies the center of the stack's window, relative to the top left corner of the main screen. (Unlike the location of controls, the location of a stack is specified in absolute coordinates.) The following statement moves a stack to the center of the main screen:

```
set the location of stack "Wave" to the screenLoc
```

The `rectangle` property of a stack specifies the position of all four edges, and can be used to resize the window as well as move it:

```
set the rectangle of this stack to "100,100,600,200"
```

Tip: To open a window at a particular place without flickering, set the stack's location or rectangle property to the desired value before going to it.

You can also use associated properties to move a window. Changing a stack's bottom or top property moves the window up or down on the screen. Changing the left or right property moves the window from side to side.

Changing a window's layer

You bring a window to the front by using the go command:

```
go stack "Alpha"
```

If the stack is already open, the go command brings it to the front, without changing its mode.

To find out the layer order of open stack windows, use the openStacks function. This function lists all open stack windows in order from front to back.

The palette layer:

Normally, palette windows float above editable windows and modeless dialog boxes: that is, a palette is always above a standard window, even if you bring the standard window to the front.

This helps ensure that palettes, which usually contain tools that can be used in any window, cannot disappear behind document windows. It's also a good reason to make sure you design palette windows to be small, because other windows cannot be moved in front of them if the palette blocks part of the window.

To change this behavior and allow palettes to be interleaved with other windows, set the raisePalettes property to false.

The system palette layer:

System windows—stacks whose systemWindow property is true—float above all other windows, in every running application. This means that even if the user brings another application to the front, your application's system windows remain in front of all windows.

System windows are always in front of other windows, and you cannot change this behavior.

The active window

In most applications, commands are applied to the active window. Since Revolution gives you the flexibility to use several different window types, not all of which are editable, the current stack is not always the same as the active window. The current stack is the target of menu choices (such as View menu Go Next) and is the stack specified by the expression this stack.

For example, executing the find command may have unexpected results if stacks of different modes are open, because under these conditions, the search may target a stack that is not the frontmost window.

Finding the current stack:

The current stack—the stack that responds to commands—is designated by the defaultStack property. To determine which stack is the current stack, use the following rules:

1. If any stacks are opened in an editable window, the current stack is the frontmost unlocked stack. (A stack is unlocked if its `cantModify` property is set to true.)
2. If there are no unlocked stacks open, the current stack is the frontmost locked stack in an editable window.
3. If there are no stacks open in an editable window, the current stack is the frontmost stack in a modeless dialog box.
4. If there are no editable or modeless windows open, the current stack is the frontmost palette.

Another way of expressing this set of rules is to say that the current stack is the frontmost stack with the lowest mode property. You can find out which stack has the lowest mode using the `topStack` function.

The `topStack` function and the `defaultStack` property:

The `defaultStack` property specifies which stack is the current stack. By default, the `defaultStack` is set to the `topStack`, although you can change the `defaultStack` to any open stack.

The `topStack` function goes through the open stacks first by mode, then by layer. For example, if any editable windows are open, the topmost editable window is the `topStack`. If there are no editable windows, the `topStack` is the topmost modeless dialog box, and so on.

Changing the current stack:

To operate on a stack other than the current stack, set the `defaultStack` property to the stack you want to target before executing the commands. (Usually, the `defaultStack` is the `topStack`, but you can change it if you want to override the usual rules about which window is active.)

A note about Unix systems:

If your system is set up to use pointer focus rather than click-to-type or explicit focus, you may experience unexpected results when using Revolution, since the current stack will change as you move the mouse pointer. It is recommended that you configure your system to use explicit focus when using Revolution or any other applications created in Revolution.

Creating a backdrop

For some applications, you may want to create a solid or patterned backdrop behind your application's windows. This backdrop prevents other applications' windows from being seen—although it does not close those windows—so it's appropriate for applications like a game or kiosk, where the user doesn't need to see other applications and where you want to keep distractions to a minimum.

To create a backdrop, you set the `backdrop` property to either a color reference, or the ID of an image you want to use as a tiled pattern:

```
set the backdrop to "#99FF66" -- a color
set the backdrop to 1943 -- an image ID
```

Tip: In the Revolution development environment, you can create a backdrop by choosing View menu>Backdrop. (Use the Preferences dialog box to specify a backdrop color to use.)

Open, Closed, and Hidden Windows

Each open stack is displayed in a stack window. A stack can be open without being visible, and can be loaded into memory without being open.

Hidden stacks

A stack window can be either shown or hidden, depending on the stack's visible property. This means a window can be open without being visible on the screen.

Tip: To list all open stacks, whether they're visible or hidden, use the openStacks function.

Loaded stacks

A stack can also be loaded into memory without actually being open. A stack whose window is closed (not just hidden) is not listed by the openStacks function. However, it takes up memory, and its objects are accessible to other stacks. (For example, if a closed stack that's loaded into memory contains a certain image, you can use the image as a button icon in another stack.)

A stack can be loaded into memory without being open under any of the following conditions:

- ò A handler in another stack referred to a property of the closed stack. This automatically loads the referenced stack into memory.

- ò The stack is in the same stack file as another stack that is open.

- ò The stack was opened and then closed, and its destroyStack property is set to false. (If the destroyStack property is false, the stack is closed, but not unloaded, when its window is closed.)

Tip: To list all stacks in memory, whether they're open or closed, use the revLoadedStacks function.

The states of a stack

A stack, then, can be in any of four states:

- ò Open and visible: The stack is loaded into memory, its window is open, and the window is visible.

- ò Open and hidden: The stack is loaded into memory, its window is open, but the window is hidden. The stack is listed in the Window menu and in the Application Browser.

- ò Closed but loaded into memory: The stack is loaded into memory, but its window is not open and it is not listed by the openStacks function or in the Window menu. However, its objects are still available to other stacks, and it is listed in the Application Browser. A stack that is closed but loaded into memory has a mode property of zero.

To remove such a stack from memory, choose Tools menu Application Browser, find the stack's name, and Control-click it (Mac OS or OS X) or Right-click it (Unix or Windows) in the Application Browser window and choose "Close and Remove from Memory" from the contextual menu.

- ò Closed: The stack is not loaded into memory and has no effect on other stacks.

Summary

In this topic, you have learned that:

ò Revolution can display any stack as an editable window, palette, modal dialog box, or modeless dialog box, or (on OS X systems) a sheet or drawer in another stack.

ò How a stack is displayed depends on what command was used to open it and on the stack's style property.

ò The stack's decorations property determines whether it has a title bar, close box, menu bar (on Windows and Unix systems), and other window parts.

ò The frontmost stack window is not necessarily the current stack for purposes of commands that operate on the current stack. Which stack is the current stack depends on the window layer and window type of open stacks.

ò A stack can be open without being visible on the screen, and can even be loaded into memory without being open.

About menus and the menu bar

See Also:

About containers, variables, and sources of value, Menu Builder Tutorial, How to assign a keyboard equivalent to a menu item, How to change a menu's contents when it is opened, How to create a separator line in a menu, How to enable or disable a menu item, How to respond to quitting an OS X application, How to show or hide the menu bar, How to simulate a menu choice, How to switch between menu bars, Why are some menu items disabled?, Why does a menu item have missing characters?, Why does the top of my stack window disappear when I add a menu bar?, Why don't the menus appear when I open a stack?, Recipe for automatically updating a Text menu, Recipe for automatically updating a Window menu, Tools menu > Menu Builder, defaultMenuBar property, disable menu command, editMenus property, enable menu command, lockMenus property, menubar property, menuHistory property, menuPick message

Menus in Revolution are not a separate object type. Instead, you create a menu from either a button or a stack, then use special commands to display the menu or to include it in a menu bar.

This topic discusses menu bars, menus that are not in the menu bar (such as contextual menus, popup menus, and option menus), how to make menus with special features such as checkmarks and submenus, and how to use a stack window as a menu for total control over menu appearance.

To easily create menu bars that work cross-platform, choose Tools menu Menu Builder. The details about menu bars in this topic are needed only if you decide to bypass the Menu Builder, if you want to include specific features it doesn't support, or if you're just curious about what's going on behind the scenes when a menu bar is created.

To fully understand this topic, you should know how to create buttons, create groups, and write short scripts. If you have gone through the *Getting Started* tutorial, you have enough information to fully understand this topic.

Contents:

- Menu Types

- Button Menus

- Creating Cascading Menus

- Special Effects in Menus

- Menu Bars on Unix and Windows Systems

- Menu Bars on Mac OS and OS X Systems

- Stack Menus

Menu Types

Revolution supports several menu types: pulldown menus, option menus (usually called popup menus on Mac OS and OS X), popup menus (usually called contextual menus on Mac OS and OS X), and combo boxes.

Each of these menu types is implemented by creating a button. If the button's style property is set to *menuItem*, clicking it causes a menu to appear. The button's *menuMode* property determines what kind of menu is displayed.

Even menu bars are created by making a pulldown-menu button for each menu, then grouping the buttons to create a single menu bar. The menu bar can be moved to the top of the stack window (on Unix and Windows systems). To display the menu bar in the standard location at the top of the screen on Mac OS and OS X systems, you set the stack's menubar property to the group's name. The name of each button is displayed in the menu bar as a menu, and pulling down a menu displays the contents of the button as a list of menu items.

Button Menus

The easiest way to create a menu is to create a button and set the style of the button to "menu". Next, you can set the menuMode of the button to the appropriate menu type. You can either set the menuMode in a handler, or use the Type menu in the button's property inspector to set the menu type.

To create the individual menu items that will appear in the menu, set the button's text property to the menu's contents, one menu item per line. You can either set this property in a handler, or fill in the box labeled "Menu items" on the Basic Properties pane of the property inspector.

When you click the button, the specified menu type appears, with the text you entered displayed as the individual menu items in the menu.

Tip: To dynamically change the menu's contents at the time it's displayed, put a mouseDown handler in the button's script that puts the desired menu items into the button. When the menu appears, it displays the new menu items.

For menus that retain a state (such as option menus and combo boxes), the button's label property holds the text of the currently chosen menu item.

Handling the menuPick message:

When the user chooses a menu item from the menu, Revolution sends the menuPick message to the button. The message parameter is the name of the menu item chosen. If you want to perform an action when the user chooses a menu item, place a menuPick handler like this one into the button's script:

```
on menuPick theMenuItem
  switch theMenuItem
    case "Name of First Item"
      -- do stuff here for first item
      break
    case "Name of Second Item"
      -- do stuff here for second item
      break
    case "Name of Third Item"
      -- do stuff here for third item
      break
  end switch
end menuPick
```

Changing the currently-chosen menu item:

For menus that retain a state (such as option menus and combo boxes), you can change the currently-chosen menu item by changing the button's label property to the text of the newly chosen item

If you change the currently-chosen menu item in option menus, also set the button's menuHistory property to the line number of the newly chosen item. This ensures that the new choice will be the one under the mouse pointer the next time the user clicks the menu.

Creating Cascading Menus

To create a cascading menu (also called a submenu, pull-right menu, or hierarchical menu), add a tab character to the start of menu items that you want to place in the submenu.

For example, the following text, when placed in a menu button, creates two menu items, then a submenu containing three more items, and finally a last menu item:

```
First Item
Second Item
Third Item Is A Submenu
    First Item In Submenu
    Second Item In Submenu
Last Menu Item Not In Submenu
```

The depth of a submenu item is determined by the number of tab characters before the menu item's name. The submenu item becomes part of the closest line above the submenu item that has one fewer leading tab character.

This means that the first line of a menu cannot start with a tab character, and any line in the button's text can have at most one more tab character than the preceding line.

Important! You cannot create a cascading combo box at all, and cascading option menus do not work properly on all platforms. In general, you should create cascading menus only as a part of a pulldown menu.

Cascading menus and the menuPick message:

When the user chooses a menu item in a cascading menu, the parameter of the menuPick message contains the menu item name and the name of the submenu it's part of, separated by a vertical bar (|). For example, if the user chooses the "Second Item In Submenu" from the menu described above, the parameter sent with the menuPick message is:

```
Third Item Is A Submenu|Second Item In Submenu
```

Special Effects in Menus

There are several special characters that you can put at the start of a line in the button's contents to change the behavior of the menu item:

- A dash on a line by itself creates a divider line
- !c checks the menu item
- !n unchecks the menu item
- !r places a diamond at the start of the menu item
- !u removes the diamond

If you include any of the above special characters in a submenu item, the special character must be placed at the start of the line before the tab characters that make it a submenu item.

Note: You cannot create divider lines in combo boxes, or in option menus on Windows systems.

There are three other special characters that can appear anywhere in a line:

ò Putting the & character anywhere in a line underlines the next character and makes it the keyboard mnemonic for that menu item on Windows systems. The & character does not appear in the menu, and is not sent with the parameter to the menuPick message when you choose the item from a menu.

ò Putting the / character anywhere in a line makes the next character the keyboard equivalent for the menu item. Neither the / nor the character following it appear in the menu, nor do they appear in the parameter to the menuPick message.

To put an & or / character in the text of a menu, double the characters: && or //.

ò Putting the (character anywhere in a line disables the menu item. To put a (character in a menu item without disabling it, precede it with a backslash: \.

Note: You cannot disable lines in combo boxes, or in option menus on Windows systems.

All of the above special characters are filtered out of the parameter sent with the menuPick message when the user chooses a menu item. The parameter is the same as the characters that are actually displayed in the menu.

Tip: The font and color of a button menu is determined by the button's font and color properties. However, on Mac OS and OS X systems, if the lookAndFeel is set to Appearance Manager, the font and color of option menus and popup menus is controlled by the operating system's settings, rather than by the button's font and color properties.

Enabling and disabling menu items:

To enable or disable a menu item in a handler, you can add or remove the (special character, but it is generally easier to use the enable menu and disable menu commands:

```
enable menuItem 3 of button "My Menu"  
disable menuItem 4 of me
```

These commands simply add or remove the (special character at the start of the designated line of the button's contents.

Menu Bars on Unix and Windows Systems

A menu bar is made up of a group of menu buttons, with the menuMode property of each button set to pullDown.

Tip: The Menu Builder can automatically create a menu bar for you. To use the Menu Builder, choose Tools menu Menu Builder.

To create a menu bar by hand without using the Menu Builder:

1. Create a button for each menu, set the style of each button to `menuItem`, and set the `menuMode` of the button to `pullDown`. You can either set these properties in a handler, or simply choose Object menu New Control PullDown Menu to create each button.
2. Put the menu items into each button's contents. In each button's script, create a `menuPick` handler to perform whatever actions you want to do when a menu item is chosen.
3. Select the buttons and form them into a group, then move the group to the appropriate position at the top of the window. For Windows systems, set the `textFont` of the group to the standard font for Windows menus, `MS Sans Serif`.

Important! The buttons in your menu bar should not overlap. Overlapping buttons may cause unexpected behavior when the user tries to use a menu.

Menu Bars on Mac OS and OS X Systems

To create a Mac OS menu bar, you follow the same steps as for a Unix and Windows menu bar. This places a group of buttons, each of whose `menuMode` property is set to `pullDown`, at the top of your stack window.

Next, you set the `menubar` property of your stack to the name of the group. This does two things: it displays the menus in the menu bar at the top of the screen, and it shortens the stack window and scrolls it up so that the group of menu buttons is not visible in the window. (Since the menus are in the menu bar, you don't need to see them in the stack window as well.)

Important! If your stack has more than one card, make sure that the group is placed on all the cards. (To place a group on a card, choose Object menu Place Group, or use the `place` command.) This ensures that the menu bar will be accessible on all cards of the stack, and prevents the stack from changing size as you move from card to card (to accommodate shortening the stack window for the menu bar group).

The default menu bar:

If other stacks in your application don't have their own menu bars, set the `defaultMenubar` global property to the name of your menu group, as well as setting the stack's `menubar` property. The `defaultMenubar` is used for any stack that doesn't have a menu bar of its own.

Tip: For a custom menu bar to work correctly inside the Revolution development environment, overriding the Revolution menu bar, you must set the `defaultMenubar` to the name of your menu group.

The layer of menu buttons:

For a menu bar to work properly on Mac OS and OS X systems, the menus must be in layer order within the group. That is, the button for the File menu must be numbered 1, the button for the Edit menu must be 2, and so on. (The Menu Builder takes care of this automatically; you only need to worry about layering if you're creating the menu bar by hand.)

Changing menus dynamically:

If you want to dynamically change a menu's contents with a `mouseDown` handler at the time the menu is displayed, you must place the `mouseDown` handler in the group's script. When a menu button is being displayed in the Mac OS menu bar, it does not receive `mouseDown` messages, but its group does.

The editMenus property:

When you set the menubar property of a stack to the name of a group, the stack is resized and scrolled up so the part of the window that holds the menus is not visible. To reverse this action so you can see, select, and edit the buttons that make up your menu bar, set the editMenus property to true. This resizes the stack window so the button menus are again visible, and you can use the tools in the Revolution development environment to make changes to them.

To scroll the stack window again so that the menus are hidden, set the editMenus property back to true.

Special menu items

A few menu items on Mac OS and OS X are handled directly by the operating system. To accommodate these special menu items while allowing you to create a fully cross-platform menu bar, Revolution treats the last two menu items of the Help menu (for Mac OS and OS X), the File menu (OS X), and the Edit menu (OS X) differently.

By following these guidelines, you can make sure your menus will appear properly on all operating systems without having to write special code or create platform-specific menu bars.

The Help menu and the “About This Application” menu item:

When Revolution sets up the Mac OS menu bar, it automatically makes the last button the Help menu (regardless of the button’s name). The standard Help menu items, such as “About Balloon Help” and “Show Balloons” on Mac OS, are included for you automatically; you don’t need to include them in your Help menu button, and you can’t eliminate them from the Help menu.

Revolution moves the last menu item in the Help menu to the “About This Application” position. (On Mac OS systems, this is the first menu item in the Apple menu. On OS X systems, it’s the first menu item in the Application menu.) Therefore, the last menu item in your Help menu button should be an appropriate “About” item. The menu item above it must be a divider line (a dash), and above that must be at least one menu item to be placed in the Help menu.

The File menu and the “Quit” menu item:

On OS X systems, the “Quit” menu item is normally placed in the Application menu (which is maintained by the operating system) rather than in the File menu as is standard on other platforms. To accommodate this user-interface standard, Revolution removes the last two menu items of the File menu when a standalone application is running on an OS X system. Therefore, the last menu item in your File menu button should be “Quit”. The menu item above it should be a divider line (a dash).

The Edit menu and the “Preferences” menu item:

On OS X systems, the “Preferences” menu item is also normally placed in the Application menu. To accommodate this user-interface standard, Revolution removes the last two menu items of the Edit menu when a standalone application is running on an OS X system. Therefore, the last menu item in your Edit menu button should be “Preferences”. The menu item above it should be a divider line (a dash).

Note: The Preferences menu item is treated in this special way only if its name starts with the string “Preferences”.

Tip: If your application's user interface is presented in a language other than English, set the name of the Edit menu button to `edit`, and set its label to the correct translation. This ensures that the engine can find the Edit menu, while making sure that the menu is shown in the correct language.

Choosing the special menu items:

When the user chooses any of these special menu items, a `menuPick` message is sent to the button that the menu item is contained in. This ensures that your button scripts will work on all platforms, even if Revolution displays a menu item in a different menu to comply with user-interface guidelines.

For example, if the user chooses `About This Application` from the Apple menu on a Mac OS system, a `menuPick` message is sent to the Help menu button, with `About This Application` as its parameter. You handle the message for the About menu item in the Help menu button's script, even though Revolution displays this menu item in a different menu on the Mac.

Stack Menus

Button menus can be used for most kinds of standard menus. However, if you want to create a menu with a feature that is not supported by button menus—for example, if you want a popup menu that provides pictures, rather than text, as the choices—you can create a menu from a stack.

Creating a stack menu:

To create a stack menu, you create a stack with a control for each menu item. Since the stack menu is a stack and each menu item is an object, the menu items receive mouse messages such as `mouseEnter`, `mouseLeave`, and `mouseUp`.

When the user chooses an item from the stack menu, a `mouseUp` message is sent to that control. To respond to a menu item choice, instead of handling the `menuPick` message, you can place a `mouseUp` handler in the script of the object.

To create a stack menu that looks like a standard menu, create a button in the stack for each menu item. The button's `autoArm` and `armBorder` properties should be set to `true`. Or you can choose `Menu Item` item in the `New Control` submenu of the Object menu to create a button with its properties set to the appropriate values.

Be sure to set the rectangle of the stack to the appropriate size for the menu. Remember, when you open the menu, the stack will be displayed exactly as it looks in an editable window.

Finally, either set the `menuName` property of a button to a reference to the stack, or place a `mouseDown` handler containing a `pullDown`, `popup`, or `option` command in the script of an object. When you click the button or object, the stack menu appears.

Displaying a stack menu:

Stack menus can be associated with a button, just like button menus. But when you click the button, instead of displaying a menu with the button's contents, Revolution displays a stack with the behavior of a menu.

You can also display a stack menu without associating it with a button, by using the `pullDown`, `popup`, or `option` command. Normally, you use these commands in a `mouseDown` handler, so that the menu appears under the mouse pointer:

```
on mouseDown -- in card script
  popup stack "My Menu Panel"
end mouseDown
```

About connecting to and using SQL databases

See Also:

Database Types Reference, How to close the connection to a database, How to create a record set in a SQL database, How to display a single record from an automatic database query, How to display records from an automatic database query as a table, How to execute a SQL statement on a database, How to find out which record sets are open in a database, How to find out which databases are open, How to get the contents of a database field, How to get the contents of records in a SQL database, How to make changes to a SQL database, How to move through the records in a record set (database cursor), How to navigate among records from an automatic database query, How to save changes to a SQL database, How to save changes to records from an automatic database query, How to set up an automatic database query, How to use a SQL query to select records in a database, Tools menu > Database Query Builder, revCloseCursor command, revCloseDatabase command, revCommitDatabase command, revCurrentRecord function, revCurrentRecordIsFirst function, revCurrentRecordIsLast function, revDatabaseColumnCount function, revDatabaseColumnIsNull function, revDatabaseColumnLengths function, revDatabaseColumnNamed function, revDatabaseColumnNames function, revDatabaseColumnNumbered function, revDatabaseColumnType function, revDatabaseConnectResult function, revDatabaseCursors function, revDatabaseID function, revDatabaseType function, revDataFromQuery function, revExecuteSQL command, revMoveToFirstRecord command, revMoveToLastRecord command, revMoveToNextRecord command, revMoveToPreviousRecord command, revNumberOfRecords function, revOpenDatabase function, revOpenDatabases function, revQueryDatabase function, revQueryDatabaseBLOB function, revQueryResult function, revRollBackDatabase command

With the Revolution Database library, your application can communicate with external SQL databases. You can get data from single-user and multi-user databases, update data in them, get information about the database structure, and display data from the database in your stack. And with the Database Query Builder, you can automate the process of querying a database and populating fields with the data, with no scripting required.

This topic discusses how to install necessary software to communicate with databases, how to set up automatic database queries using the Database Query Builder, and how to use the Database library to communicate between Revolution and a database. (This topic does not include discussion of how to set up and create a SQL database, which is beyond the scope of the Revolution documentation.) To fully understand this topic, you should know how to write short scripts and should understand the basic concepts of SQL databases (rows and columns, database cursors, and SQL queries).

Note: A few terms used in this topic, such as `field` and `cursor`, are part of the standard terminology for working with databases, but have a different meaning in the context of Revolution development. When going back and forth between database work and more general application development, be sure you understand which meaning is applicable in the context you're currently working in. When referring to database-specific terms, the documentation usually uses phrases like `database field` or `database cursor` to remind you of the context. You can click these phrases to see their definitions.

Contents:

Introduction to Database Access

Choosing a Database
Software for Database Access
The Database Query Builder
Using the Database Library
Building Standalone Applications
Summary

Introduction to Database Access

A database is an external resource that holds information, structured in a special form for quick access and retrieval. Databases can be:

- ò any size from small to extremely large
- ò located on the same system as the application or on a remote server
- ò accessible by one user at a time or by many users at once

SQL databases

A SQL database is a database that you access and control using SQL, a standard database-access language which is widely supported. You use SQL queries (statements in the SQL language) to specify the part of the database you want to work with, to get data, or to make changes to the database.

RevolutionÆs database access is fully-featured. You can send any SQL statement to a database. You can open multiple databases (or multiple connections to the same database), maintain multiple record sets (database cursors) per connection, and send and receive binary data as well as text. You can do all this using the Database Query Builder, or in scripts that use the commands and functions in the Database library.

Tip: To see a list of Transcript terms in the Database library, open the Documentation window, click Transcript Dictionary, and choose ôDatabase Libraryö from the menu.

Why use an external database?

You donÆt need to use an external database to store information for your application. You can store information in stacks, in text files, and in a variety of other files, read it into your application as needed, and modify it.

However, in some circumstances an external database offers many advantages. A database located on a server can be accessed by more than one user. Depending on the particular implementation, a database may be suitable for constant access by hundreds of users, each getting and updating data continually. SQL databases have built-in record-locking capabilities, preventing one userÆs changes from wiping out anotherÆsàa necessity for reliable multi-user databases. Other built-in features work behind the scenes to make sure the data in the database is not corrupted by interference between different users.

SQL databases are also built for speed. When searching tens or hundreds of megabytes, or more, the performance of an optimized database will generally be much better than that of a stack doing the same search. Moreover, stacks must be loaded into memory to be searched, and therefore the whole collection of data must fit in local memory.

Finally, if you use an external database, you can put the heavy-duty processing on a server designed for the purpose, while using RevolutionÆs flexibility to give the user options for selecting data, then presenting it in a usable form.

The basics of database structure

A database is built of records, which in turn are built out of database fields. A field is the smallest part of a database that can be separately addressed. Each database field contains a particular kind of information. This might be a name, a file path, a picture, or any other kind of information. Each record contains one value for each of its fields. A set of records is called a database table, and one or more tables comprise a database.

Here's an example: suppose you have a database of customers for your business. The fields of this database might include the customer name, a unique customer ID number, and shipping address. Each record consists of the information for a single customer, so each record has a different customer name, shipping address, and so on.

Tip: You may have noticed that the database structure being described resembles a multiple-card stack that has the same fields on each card. A database field is like a field in a stack, and a record is like a card. A stack set up this way can act as a database, in fact, but lacks some of the features of an external database, such as the ability to perform SQL queries and the ability to perform robustly when accessed by more than one user.

You can also think of the set of customer records as a grid (like a spreadsheet). Each row is a record, and each column is a field, so each cell in the grid contains a different piece of information about a particular customer. Here's an example:

ID	Customer Name	Address	Country
123	Jane Jones	234 E. Street	U.K.
836	Acme Corporation	PO Box 23788	USA
823	CanCo, Inc.	1 CanCo Blvd.	Japan

There are three rows in this grid (each is the record for a particular customer) and four columns (each is one of the fields).

A row of the database means one single customer record, which has one value for each field. A column of the database means the set of all values for one of the fields, one for each record (for example, the set of all customer addresses).

More generally, each row describes one of the things in the database, and each column describes a particular state for each thing in the database.

The set of all customer records makes a table. Your database might include only this table, or it might include other related tables, such as a list of all sales. You can also connect related data from different tables of the same database. For example, if each record in the Sales table includes the ID number of the customer who bought the product, you can link all the sales records for a customer to that customer's record in the Customers table.

SQL and record sets (database cursors)

SQL works primarily with sets of records, rather than individual rows. When you send a SQL query to a database, the query typically selects certain records which you can perform further operations on. The set of records resulting from a SQL query is called a database cursor, or record set. SQL queries let you

describe the characteristics of the records you require, instead of processing each record one by one to find out whether it matches your criteria.

For example, consider the customer table we talked about in the previous section. To work with only US customers, you can write a SQL query that selects only records where the country field is 'USA'. This subset of records then becomes a record set. You can find out more about the fields and records contained in this record set, and move from record to record within this subset of the database. You can create more than one record set to work with more than one set of records at a time.

Note: Different database implementations have different limitations on movement within a record set. For example, some databases won't let you move backward within a record set: instead, you must start at the first record and move forward in order to examine the data.

Choosing a Database

Revolution directly supports the following database implementations:

- ò Oracle
- ò MySQL
- ò PostgreSQL
- ò Valentina

Revolution also supports connecting to a database via ODBC. You can use ODBC to use Access, FileMaker, and many other database implementations. (See below for more information about ODBC.)

Transcript's database commands and functions use the same syntax regardless of what type of database you are connecting to. You don't need to learn a separate database language for each type. Instead, when you first open a database with the `revOpenDatabase` command, you specify the type as one of the parameters so Revolution knows what type of database it's dealing with. The Database library handles the details of each type behind the scenes for you.

When you use the Database Query Builder, you simply select the database type you want to use. Like the Database library, the Database Query Builder handles the details for you. You can easily switch between database types.

Reasons to choose a database type

Which type of database to choose depends on a number of factors. If you need to work with an existing database, or already have a database manager installed, the decision is made for you. Likewise, if you're already an expert at a particular database implementation, you'll probably prefer to go on using that one.

Another factor is your edition of Revolution. Revolution Enterprise is the only edition that can access Oracle databases directly (not via ODBC), so if you're using another edition, you will probably choose MySQL, PostgreSQL, or Valentina.

Other factors in your choice may include price, performance, licensing model (commercial or open source), and platform support: PostgreSQL does not support Mac OS, and Valentina does not support Unix. If your users are on a particular set of platforms, you'll need to choose a database that is supported on all those platforms.

Overview of ODBC

Open Database Connectivity (ODBC) is a system that allows developers to access any type of compatible database in a standard way.

To communicate with a database, you usually have to add code that uses the database's own proprietary protocols. Without ODBC, in order to create a program that can communicate with—for example—FileMaker, Access, and Oracle databases, Revolution would have to include code for three different database protocols. With ODBC, Revolution can communicate with any of these database types using the same code.

ODBC managers:

To work with databases through ODBC, you need two pieces of software: an ODBC manager, plus a database driver for the specific database type you're using.

Windows 2000 and Windows XP, and OS X version 10.2 and later, include ODBC software with the operating system. For earlier versions, and for Mac OS and Unix systems, you can download an ODBC manager and a set of drivers. (See the section below titled "Software for Database Access" for more information.)

Performance for direct access versus ODBC access:

Typically, accessing a database via ODBC takes more configuration and is slower than accessing the database directly. For this reason, Revolution provides the ability to access MySQL, PostgreSQL, and Valentina databases (and, for Revolution Enterprise, Oracle databases) directly without going through the ODBC protocol. This ability will be valuable for anyone doing complex or extensive professional database work.

The syntax of the functions in the Database library is identical for all database types, so you do not need to rewrite your scripts to take advantage of the increased efficiency of direct access.

Valentina databases and the Valentina engine

Valentina is a fast, efficient single-user database engine for Mac OS, OS X, and Windows systems. (It is not currently available for Unix systems.)

For detailed information about Valentina, visit the Paradigma Software web site at <http://www.paradigma.com/product/vxcmd.html>. To find out more about how Valentina works with Revolution, visit the Runtime Revolution web site at <http://www.runrev.com/revolution/info/moreinformation/valentina.html>.

Valentina demonstration limits:

A demonstration version of Valentina (with a ten-minute timeout) is included in the Revolution distribution for each of the supported platforms. If you have installed Revolution on Mac OS, OS X, or Windows, you also have a demo version of Valentina installed and ready to try out. Unlike MySQL, Oracle, or PostgreSQL, Valentina doesn't require additional database server software, so it is all you need to create and communicate with a database on your system.

To remove the ten-minute timeout from the demo, license the Valentina database engine from Paradigma Software by going to the Valentina page at <http://www.runrev.com/revolution/info/moreinformation/valentina.html>. When you license Valentina, you receive a serial number. To bypass the timeout, you provide this serial number when connecting to a Valentina database. (See the Transcript Dictionary entry for the `revOpenDatabase` function for details.)

The Valentina VXCMD:

Valentina is available in several forms. Revolution uses the Valentina VXCMD technology; the Valentina engine is implemented as an external. You can also access the VXCMD directly (as described in the Valentina documentation available from Paradigma) without using the Database library or the Database Query Builder. Using Revolution's built-in Valentina access offers these advantages over directly calling the Valentina external from your scripts:

- ò The Database library syntax can be used for all types of databases, so there is no need to make code changes for Valentina databases.

- ò You can pass and retrieve binary data using the `revQueryDatabase`, `revQueryDatabaseBLOB`, and `revDataFromQuery` functions.

- ò The Database library fully supports binding variable values to a SQL query using the `?:number` syntax. (See the Transcript Dictionary entry for the `revQueryDatabase` function for details.)

Differences between editions of Revolution

The level of database access you can use depends on your edition of Revolution.

The Enterprise edition includes full access to all Revolution's database capabilities for MySQL, Oracle, PostgreSQL, and Valentina databases. It also includes full access via ODBC.

Other editions include full access to Revolution's database capabilities for MySQL, PostgreSQL, and Valentina databases, as well as full access via ODBC, but do not include direct access to Oracle databases.

The same differences apply to applications you create. Applications created with the Enterprise edition have access to all database types, including Oracle databases, while applications created with other editions do not have Oracle access.

Note: Revolution distributions include a demonstration version of Valentina. To use Valentina databases for longer than ten minutes, you must license the Valentina VXCMD software separately. However, the Database library and Database Query Builder themselves impose no restrictions on use of Valentina databases.

Software for Database Access

To provide connectivity to databases, Revolution works with database drivers—software that translates application requests into the protocol required by a specific database.

Finding database drivers

Database drivers for certain database types are included in the Revolution distribution. (The list of included database types depends on the platform.) The sections below list which database drivers are included with which platforms.

If you have installed Revolution, you have all the software needed to use the included database types. For other database types, you will need to obtain the appropriate database drivers before you can work with those databases.

Note: This section includes links to third-party web sites and contains information about third-party software. This information is provided for your convenience, but Runtime Revolution is not responsible for the software packages and sites referenced.

MySQL:

MySQL database drivers are included as part of the Revolution installation on Mac OS, OS X, and Windows systems.

To download an appropriate MySQL driver for your flavor of Unix, visit the MySQL web site at [<http://www.mysql.com/downloads/>](http://www.mysql.com/downloads/).

Oracle:

Oracle database drivers are not included as part of the Revolution installation on any platform.

To obtain an Oracle database driver for your platform, visit the Oracle web site at [<http://www.oracle.com>](http://www.oracle.com).

PostgreSQL:

A PostgreSQL database driver is included as part of the Revolution installation on Windows systems.

To download an appropriate PostgreSQL driver for OS X or Unix systems, visit the PostgreSQL web site at [<http://www.postgresql.com>](http://www.postgresql.com).

Note: PostgreSQL is not supported on Mac OS systems.

Valentina:

A demonstration version of the Valentina engine is included as part of the Revolution installation on Mac OS, OS X, and Windows systems. To find out how to license Valentina and remove its demonstration limits, visit the Runtime Revolution web site at [<http://www.runrev.com/revolution/info/moreinformation/valentina.html>](http://www.runrev.com/revolution/info/moreinformation/valentina.html).

Note: Valentina is not supported on Unix systems.

ODBC managers and database drivers

To use a database via ODBC, you must install the necessary ODBC software for your platform. (Some operating systems include an ODBC installation.) ODBC software includes one or more database drivers, plus an ODBC manager utility.

ODBC on Mac OS systems:

Revolution supports the DataDirect ODBC software. Versions exist for Mac OS 8.0 and later. You can download a 30-day trial copy from [<http://www.metrotechnologies.com/dvpages/products/datadirect.cfm>](http://www.metrotechnologies.com/dvpages/products/datadirect.cfm).

ODBC on OS X systems:

OS X version 10.2 (Jaguar) and later includes iODBC software as part of the standard system installation. To configure ODBC on OS X systems, use the ODBC Administrator application in the Utilities folder.

ODBC on Windows systems:

Windows 2000 and Windows XP include the MDAC (Microsoft Data Access Components) package as part of the standard system installation. To configure ODBC on Windows systems, use the ODBC Data Sources control panel.

For earlier versions of Windows, you can download MDAC from the Microsoft web site at [<http://www.microsoft.com/data/download.htm>](http://www.microsoft.com/data/download.htm).

ODBC on Unix systems:

Revolution supports iODBC and UnixODBC on Unix systems. You can download the iODBC software from the iODBC web site at [<http://www.iodbc.org/>](http://www.iodbc.org/). You can download the unixODBC software from the unixODBC web site at [<http://www.unixodbc.org>](http://www.unixodbc.org).

Creating a DSN for ODBC access

Once you have installed the necessary software, you use the ODBC manager to create a DSN, which is a specification that identifies a particular database.

You use the DSN to connect to the database via ODBC. The following example opens a connection to a database whose DSN is named `myDB`:

```
get revOpenDatabase("ODBC","myDB","jones","pass")
```

To connect to a database using the Database Query Builder, you enter the DSN's name in the Database Query Builder window.

Note: One of the advantages of setting up a DSN is that if you wish to change the location of the database, you only have to edit the DSN settings, not your application code. You can think of a DSN as a kind of shortcut or alias to your database.

The Database Query Builder

The Database Query Builder is a tool you use to create automatic database queries, which can be used to display data from a database in fields, images, and checkboxes. You can use an automatic query to select a record set (database cursor), display the data, navigate between records in the record set, and save changes back to the database.

Setting up and using an automatic database query requires no scripting, so the Database Query Builder is simpler to use than the commands and functions in the Database library. With the Database Query Builder, you can easily create form-based front ends to any supported database.

How the Database Query Builder works

First, you use the Database Query Builder to create an automatic query, which is stored in the stack. Then you use the Database pane in the property inspector to:

- ò automatically show the data from the query in fields and images
- ò navigate through the data using buttons
- ò update the database when the user changes the data

An automatic query can connect to the database and update the data whenever required, without any intervention by the user.

Query objects:

When you create an automatic query in the Database Query Builder, you specify all the settings needed to connect to the database and get the data you want to work with. These settings are stored in the stack, in a query object.

Query objects are implemented as groups that are not placed on any card, and therefore aren't visible to the user. The settings are stored in the custom properties of the query object.

Note: You normally work with a query object only in the Database Query Builder window, and you can usually ignore the fact that it's implemented as a group. However, if you use a script with a repeat loop that goes through all of a stack's backgrounds, be sure to skip any backgrounds that are query objects. All query objects have a custom property named `cRevGeneral["databaseQueryObject"]` that is set to true, so you can check this property to find out whether a background is a query object.

You can create as many query objects as you want. For example, you might create several query objects to get different record sets from the same database, and display the different record sets on different cards.

Tip: To make sure no one can access the settings you specify in the Database Query Builder, when you build an application, check the "Encrypt with password" option on the Stacks tab of Step 3 in the Distribution Builder. Since the settings are stored in custom properties—which are encrypted when you choose this option—no one can use a disk editor utility to find the settings.

The Database Query Builder and the Database library:

Revolution's Database Query Builder is based on the commands and functions in the Database library. This means that the Database library must be present in order for the Database Query Builder to work.

The Database Query Builder and the Database pane in the property inspector use the Database library to work with the database, handling all the details for you. You don't need to understand the commands and functions in the Database library to create and use automatic database queries.

Important! Because the Database Query Builder uses the Database library, when you build a standalone application that uses automatic database queries, make sure to check the "Database Library" option on the Inclusions tab in Step 3 of the Distribution Builder.

Setting up an automatic database query

You set up an automatic query in the Database Query Builder. To create a new query, follow these steps:

1. Choose Tools menu Database Query Builder.

Important! Because the Database Query Builder works with the query objects in the current stack, make sure the stack where you want to use the query is the current stack.

2. Click the New () button at the top of the window. A new automatic query is created.

Database connection settings:

The settings on the Connection tab control the database's connection details.

3. Give the query a name. This name should be distinctive, because you'll select automatic queries by name when setting up objects to display the data.
4. Choose the type of database you want to use and fill in the connection information. (You must set the database type first, because some of the other items in the Connections tab change depending on which type of database you're connecting to.)
5. Click Connect to connect to the database. The Database Query Builder needs to connect to the database in order to get information about its structure. This information is displayed in the Record Set tab.

Note: You can create more than one query to connect to the same database. For example, you might create several queries that fetch different record sets from the database.

Entering a SQL query:

The settings on the Record Set tab control which records from the database will be selected for the automatic query you're creating. After connecting to the database, click the Record Set tab to set up the SQL query that selects a record set to work with.

6. Choose a table from the Table menu. (If the menu is blank, go back to the Connection tab and make sure the Database Query Builder is connected to the database.)

If the database contains picture data that you'll display in an image, and this data may be larger than 64K for any one record, check the box labeled "Display large binary data".

A query that selects all the records in the table is automatically placed in the SQL Query box. To select a different set of records, enter a different SQL query in the box, then click "Refresh query". (The query must be a valid SQL SELECT statement.)

You can create as many automatic queries as you want. If two or more automatic queries have the same connection settings, they use the same connection to exchange data with the database, so you can set up several automatic queries to work with various record sets in a database without using more than one connection.

Tip: To create a new automatic query to work with the same database, it's convenient to duplicate an existing query with the required settings, then change only the SQL Query box. To duplicate an existing automatic query, choose the query you want to duplicate from the Query menu at the top of the Database Query Builder, then click the Duplicate () button.

Caching the record set:

If you check the box labeled "Cache record set", the Database Query Builder stores the data from the record set in memory, and automatically closes the record set. This means it doesn't have to contact the database every time you display a new record. Instead, it looks up the record in its stored cache.

Caching the record set is particularly useful if you are working with a database implementation that doesn't allow forward-moving database cursors (that is, that doesn't allow you to go backward from the current record to a previous record).

Note: If the record set you're working with is very large, leave the **Cache record set** box unchecked. Since the cached record set is stored in memory, caching a record set that's so large that it crowds the available memory may cause unexpected results.

Specifying the primary key:

A primary key is a database field, or combination of fields, that is used to uniquely identify each record in the table you're working with. Database records, unlike cards in a Revolution stack, don't have names or IDs, so you use the data in the primary key to identify each record.

To indicate the primary key, choose a database field from the Primary Key menu on the Record Set tab.

Note: The primary key field must be included in the SQL query (in the **SQL Query** box). If the primary key field is excluded by the query, you cannot select it as a primary key.

Databases often include a field that is specifically set up as the primary key. If you're not sure which field should be used as the primary key, check with the database administrator or with the person who created the database.

In some cases, no single database field is used as the primary key. Instead, a combination of fields is used: no single field is different for each record, but a combination of two or more fields is different for each record. To use a combination of database fields as the primary key, enter the field names, separated by commas, in the Primary Key box. (Since you're using more than one field, you cannot use the menu. Make sure to spell the field names correctly.)

Important! If you plan to update the database with information from your application, it is essential to choose a primary key that is different for each record. If you don't specify a primary key, or choose one that is the same for two or more records, updating the database will probably corrupt the data.

Connecting to a database automatically:

In the Connection tab, you can choose to have the Database Query Builder connect automatically either when the card opens or when the stack opens. Which one to select depends on how you set up the objects that display the data from the query.

If the objects that display data from the query appear only on one card, choose **Card opens** from the menu labeled **Open connection when**. In this case, the Database Query Builder automatically connects to the database when you go to a card with objects that use the query.

If the objects that display data from the query appear on all cards in the stack (for example, if you create a shared group to hold the objects), choose **Stack opens** instead. In this case, the Database Query Builder automatically connects to the database when you open the stack that contains the query.

Set the menu labeled **Close connection when** to the same setting you used for the **Open connection when** menu.

Displaying data from an automatic query

Once you've set up an automatic database query in the Database Query Builder, you can set up a field to display the data from the current record or from all the records at once. You can also display the content of boolean database fields as checkbox buttons, and display picture data in images.

To set up an object to display data from a database, you use the Database pane in the object's property inspector.

Displaying a database field in a field:

To display the information from a database field in a field, follow these steps:

1. Open the field's property inspector and choose "Database" from the menu at the top of the inspector palette.
2. From the Query menu in the Database pane, choose the automatic database query you want to use. (You must have previously created the automatic query in the Database Query Builder.)
3. From the Column menu, choose the name of the database field you want to display.

The Database Query Builder automatically gets the data from the database field of the current record and puts it into the field. (See "Controlling the database" below to find out how to change which record is the current record.)

The data is fetched again every time the field is displayed, so if the database has been updated, the field displays the updated data automatically.

Displaying records in table format:

In addition to displaying a database field from the current record, you can display all the records at once, one per line. For example, you can set up one card to show a list of all records, which the user can click to view a specific record displayed on another card.

To display the entire record set from an automatic database query, follow these steps:

1. Open the field's property inspector and choose "Database" from the menu at the top of the inspector palette.
2. From the Query menu in the Database pane, choose the automatic database query you want to use. (You must have previously created the automatic query in the Database Query Builder.)
3. From the Column menu, choose "Show All".

The Database Query Builder automatically gets the entire record set and puts it into the field. Records are separated by return characters, and database fields within a record are separated by tab characters, so the record set appears as a grid: each record is a row and each database field is a column.

As with fields that display a single database field, the data is automatically fetched again every time the field is displayed.

4. To control the way the rows and columns are displayed, choose "Table" from the menu at the top of the property inspector, then check the box labeled "Table object".

Tip: If "Show All" is selected, the field displays all the database fields specified by the automatic query's SQL query. To display a different set of fields, create a new automatic query in the Database Query Builder, with the same connection settings and the same table, but with a SQL query that

specifies the set of database fields you want to show, and use that automatic query to display all the records.

Displaying a boolean field as a checkbox:

If one of the database fields in an automatic query's record set is boolean—that is, if the value in the field can be either zero or 1, indicating false or true—you can display the field's data in a checkbox button. If the database field's content in the current record is zero (false), the checkbox is unchecked. If the content in the current record is 1 (true), the checkbox is checked.

To display the information from a boolean database field as a checkbox, follow these steps:

1. Create a checkbox button, open the button's property inspector, and choose "Database" from the menu at the top of the inspector palette.
2. From the Query menu in the Database pane, choose the automatic database query you want to use. (You must have previously created the automatic query in the Database Query Builder.)
3. From the Column menu, choose the name of the database field you want to display.

Displaying binary data in an image:

If one of the database fields in an automatic query's record set consists of binary data in one of the picture formats that Revolution is compatible with, you can display the field's data in an image.

To display the information from a database field in an image, follow these steps:

1. Open the image's property inspector and choose "Database" from the menu at the top of the inspector palette.
2. From the Query menu in the Database pane, choose the automatic database query you want to use. (You must have previously created the automatic query in the Database Query Builder.)
3. From the Column menu, choose the name of the database field you want to display.

Important! If the picture data you're using may be larger than 64K for any one record, make sure to check the box labeled "Display large binary data" on the Record Set tab in the Database Query Builder.

Creating a form that shows a single record:

If you place more than one object that displays a database field on a card, all the objects display the data from the current record. This means that you can design a form to show all the database fields you want to display by placing the appropriate objects on a card.

For example, to display an employee database, you might create fields for the employee's first name, last name, telephone extension, and department; a checkbox button to show whether the employee is exempt from overtime or not; and an image to hold the employee's photograph. When you use the Database pane in each object's property inspector to specify a query and database field, all the objects show the data from the current record.

Of course, you can add other objects that don't show database fields, such as label fields, line graphics, and buttons. By combining the Database Query Builder's ability to display data with Revolution's other user-interface controls, you can design any type of form you want.

Automatically updating the database:

So far, this topic has described the process for displaying data from a database. To enable a user to edit displayed data in an object and update the central database with the changes you check the box labeled "Update after editing" in the Database pane of the object's property inspector.

For example, if the "Update after editing" box is checked for a field, then when the user changes the text in the field, the Database Query Builder sends the new content to the database, where it replaces the original content. Similarly, if the box is checked for a checkbox button, the data in the database is changed when the user clicks the checkbox to check or uncheck it. If the box is checked for an image, the data is updated when the user paints in the image with the paint tools.

Important! If you check the "Update after editing" box for any object that displays data from an automatic database query, you must designate a primary key for the automatic query. You choose a primary key on the Record Set tab of the Database Query Builder. It is essential to choose a primary key that is different for each record. If you don't specify a primary key, or choose one that is the same for two or more records, updating the database will probably corrupt the data.

Controlling the database

Once you've set up objects to display data from an automatic database query, you can create buttons to control the database. The Database Query Builder provides pre-scripted actions for database control, so all you need to do is choose an automatic query and an action.

The Database pane of a button's property inspector (for button types other than checkboxes) provides options to move through the record set, fetch the data again from the database, or update the database.

Note: Adding a database action to a button does not change the button's script. Instead, a frontScript intercepts the mouseUp message and performs the action, then passes the message. You can create handlers in the button's script without interfering with the database action.

Navigating through a record set:

You can use a button to go to the first, last, previous, or next record in the record set. Navigating to a record makes it the current record, so all objects on the current card that display a database field are updated to show the new record.

(Fields where "Show All" is selected in the Database pane are not updated when you change records, because such fields show all the records in the record set, instead of showing the current record.)

To create a navigation button, follow these steps:

1. Create a button (with any style except "checkbox" or "radioButton"). Then open the button's property inspector and choose "Database" from the menu at the top of the inspector palette.
2. From the Query menu in the Database pane, choose the automatic database query you want to use. (You must have previously created the automatic query in the Database Query Builder.)

3. From the Action menu, choose **Move to First Record**, **Move to Next Record**, **Move to Previous Record**, or **Move to Last Record**.

Refreshing the query:

You can use a button to fetch the data from the database again, refreshing any objects that display data from the database. Refreshing the automatic query ensures that the data shown is completely up-to-date and reflects any recent changes others may have made to the database.

To create a refresh button, follow these steps:

1. Create a button (with any style except **checkbox** or **radioButton**). Then open the button's property inspector and choose **Database** from the menu at the top of the inspector palette.
2. From the Query menu in the Database pane, choose the automatic database query you want to use. (You must have previously created the automatic query in the Database Query Builder.)
3. From the Action menu, choose **Refresh SQL**.

Note: A refresh button does the same thing as the **Refresh query** button on the Record Set pane of the Database Query Builder.

Updating the database:

You can use a button to update the current record of the database. When the user clicks the button, the Database Query Builder sends the data from all objects on the current card that display a database field to the database.

To create an update button, follow these steps:

1. Create a button (with any style except **checkbox** or **radioButton**). Then open the button's property inspector and choose **Database** from the menu at the top of the inspector palette.
2. From the Query menu in the Database pane, choose the automatic database query you want to use. (You must have previously created the automatic query in the Database Query Builder.)
3. From the Action menu, choose **Update Record**.

As mentioned previously, you can set up an object to update the database automatically when the user changes the object's content. If all the objects the user can change are set to update automatically, you don't need to create an update button.

However, under some circumstances, it is convenient to leave the **Update after editing** box unchecked, and provide this capability in a separate button instead. You may want to validate the content of a field before you allow the database to be updated. (For example, if the user can edit a telephone number, you might want to check the format of the number to make sure it's a valid phone number.) Or you may need to do more complex validation tasks. In this case, you can perform validation in `closeField` handlers in the fields, then create an update button to update the entire record at once.

Important! If you create an update button for an automatic database query, you must designate a primary key for the automatic query. You choose a primary key on the Record Set tab of the Database

Query Builder. It is essential to choose a primary key that is different for each record. If you don't specify a primary key, or choose one that is the same for two or more records, updating the database will probably corrupt the data.

Using the Database Library

The Database library includes commands and functions to support all aspects of working with a SQL database.

While the Database Query Builder uses the Database library, you don't have to understand the details of the Database library to use the Database Query Builder. Use the Database library for more advanced database work, or if you need functionality that the Database Query Builder does not provide.

Tip: To see a list of all the terms in the Database library, open the Documentation window, click Transcript Dictionary, and choose "Database Library" from the menu at the top of the window.

Working with the Database library

You use the commands and functions in your scripts to control the database, and use your application's user interface to display the data and to let the user enter information.

A typical database session includes the following steps:

1. Connecting to the database:

To open a connection to the database, you use the `revOpenDatabase` function. This function returns a database ID number, which you can use to refer to the database in other commands and functions.

2. Selecting records with a SQL query:

To select the set of records you want to work with, you specify the characteristics of those records in the form of a SQL query, using the `revQueryDatabase` function. This function returns a record set ID number, which you can use to refer to that record set.

Tip: If your records may contain more than 64K of binary data, use the `revQueryDatabaseBLOB` function instead.

3. Navigating within the record set:

You can move among the records you've selected with the `revMoveToFirstRecord`, `revMoveToLastRecord`, `revMoveToNextRecord`, and `revMoveToPreviousRecord` commands.

4. Getting data from the selected records:

To get the data from a field in the current record, use the `revDatabaseColumnNamed` or `revDatabaseColumnNumbered` function.

Tip: You can get all the data from the set of records specified by a SQL query without creating a record set using the `revDataFromQuery` function.

5. Making changes to the database:

To change data in the database, you issue SQL statements using the `revExecuteSQL` command. To save (commit) changes, use the `revCommitDatabase` command, and to discard (roll back) changes, use the `revRollBackDatabase` command.

6. Closing the database connection:

To close the connection to the database and end the session, you use the `revCloseDatabase` command.

Database library commands and functions

The Transcript terms in the Database library fall into several categories. Click any term to see its entry in the Transcript dictionary.

Connecting and disconnecting:

`revOpenDatabase` function: Connect to a database

`revOpenDatabases` function: Which databases have open connections?

`revCloseDatabase` command: End a session with a database

`revSetDatabaseDriverPath` command: Where are the database drivers?

Getting information and controlling the database:

`revExecuteSQL` command: Execute SQL statements

`revCommitDatabase` command: Save changes

`revRollBackDatabase` command: Throw away changes

`revDatabaseType` function: Is this database MySQL, Oracle, etc?

`revDatabaseConnectResult` function: What was the last connection error?

`revDatabaseCursors` function: What are this database's open record sets?

Selecting records to work with:

`revQueryDatabase` function: Select records to make a record set

`revQueryDatabaseBLOB` function: Select records with binary data

Getting information and controlling a record set:

`revNumberOfRecords` function: How many records in this record set?

`revDatabaseColumnCount` function: How many fields in each record?

`revDatabaseColumnLengths` function: What's the maximum field size?

`revDatabaseColumnNames` function: What are the fields named?

`revDatabaseColumnType` function: What are the field data types?

`revDatabaseID` function: Which database is this record set in?

`revQueryResult` function: What was the last query error?

`revCloseCursor` command: Dispose of a record set

Moving between records in a record set:

`revCurrentRecord` function: Which record are we on?

`revCurrentRecordIsFirst` function: First record in the record set?

`revCurrentRecordIsLast` function: Last record in the record set?

`revMoveToFirstRecord` command: Make first record the current record

`revMoveToLastRecord` command: Make last record the current record

`revMoveToNextRecord` command: Move forward one record

`revMoveToPreviousRecord` command: Move back one record

Getting data:

`revDataFromQuery` function: Get data without making a record set

`revDatabaseColumnName` function: Get a field in the current record

`revDatabaseColumnNumber` function: Get a field in the current record

`revDatabaseColumnIsNull` function: Is a field set to null (no value)?

Note: The syntax used by the Database library is the same on all platforms, and (as far as possible) for all databases. Since some databases have more features than others, however, some commands and functions in the Database library may not be supported by all databases.

Integrating the Database library with the Database Query Builder

As mentioned above, the Database Query Builder uses commands and functions from the Database library. You can use the Database library to work with databases opened by the Database Query Builder.

Working with a database requires you to specify the database ID, assigned when the connection to the database was opened. Working with a record set requires you to specify the record set ID, assigned when the record set was fetched from the database.

Using commands and functions with a database connection:

All Database library commands and functions (except `revOpenDatabase`) require a database ID. This ID number is returned by the `revOpenDatabase` function when you connect to a database.

The Database Query Builder uses the `revOpenDatabase` function when it connects to a database, and stores the resulting database ID. To find out the database ID corresponding to an automatic database query, use a statement like the following:

```
put revConnectionOfQuery("My Query Name") into myDatabaseID
```

You can use this ID number with Database library commands and functions to control the database or get information about it.

Using commands and functions with a record set:

The Database library commands and functions that work with a record set require a database ID. This ID number is returned by the `revQueryDatabase` or `revQueryDatabaseBLOB` function when you query a database.

The Database Query Builder uses the `revQueryDatabase` function (or `revQueryDatabaseBLOB` if you check the "Display large binary data" box on the Record Set tab in the Database Query Builder) when it connects to a database or refreshes the query, and stores the resulting record set ID. To find out the record set ID corresponding to an automatic database query, use a statement like the following:

```
put revCursorOfQuery("My Query Name") into myRecordSetID
```

You can use this ID number with Database library commands and functions to get information about the record set or update it.

Building Standalone Applications

Revolution/Es database access is provided for each platform in the form of an external, plus script libraries to support automatic database queries.

To include database access in your application—using either the Database Query Builder, the Database library, or both—make sure to check the "Database Library" option on the Inclusions tab in Step 3 of the Distribution Builder window when building the application.

Database driver installation

Users of the application must have the necessary database driver installed on their system. If your application uses a database driver that is distributed with Revolution, you may include it with your own applications. Otherwise, you must either obtain the necessary database driver and license it (if necessary) for distribution with your application, or make sure it is already installed on your users' systems, or advise users where they can obtain it.

On Windows systems, the Distribution Builder places the files `libmysql.dll` (for MySQL) and `libpq.dll` (for PostgreSQL) in the `drivers` folder when you create an application. If your application uses MySQL or PostgreSQL, and the necessary database driver are not already installed on the system, your installer should move the needed file from the `drivers` folder to the folder your application is in. This enables the application to find and load the database driver when it starts up.

Standalone applications and the Database library

The Database library is provided for each platform in the form of an external. If you check the `Database Library` option in the Distribution Builder, the necessary external to support database access in your standalone will be included.

On Mac OS systems, the external resource is placed in the standalone application file. On OS X systems, the external is placed in the application bundle. On Unix and Windows systems, the external is installed in the distribution's `data` folder (which is set in the `Create` folder for `substack` option in Step 2 of the Distribution Builder), and the externals property of the application's main stack is set to include the database external file.

Standalone applications and the Database Query Builder

If you check the `Database Library` option in the Distribution Builder, all the supporting scripts for the Database Query Builder are included in the application (although the Database Query Builder window is not), along with the Database library external. You don't need to do anything more to make automatic database queries work in your application.

Summary

In this topic, you have learned that:

- ò You can use Revolution to connect to and use external databases that use the SQL query language. Revolution's database access works the same way, regardless of what database implementation you use.
- ò To use a database, you must install the necessary database driver. Revolution includes database drivers for some database types.
- ò You use the Database Query Builder to create automatic database queries. With an automatic database query, your application can automatically use a SQL query to get records from a database, display the data in objects, and automatically update the database when the user makes changes, all without writing scripts.
- ò With the commands and functions in the Database library, you can control every aspect of working with a SQL database in detail.

ò To build a standalone application that uses any form of database access, check the ôDatabase libraryö box in the Distribution Builder. This includes all necessary scripts for automatic database queries, as well as the Database library itself. Make sure the applicationÆs users have the needed database driver installed on their systems.

File

See Also:

The File menu contains commands to open, close and save files; print; and incorporate files into your stack.

New Mainstack

See Also:

File menu > New Substack of (main stack name), File menu > Open Stack..., Object menu > New Card, Object menu > New Control, About main stacks, substacks, and the organization of a stack file, create stack command

On menu File

Creates a new untitled stack window. When you save the stack, Revolution asks for a file name and location.

New Substack of (main stack name)

See Also:

File menu > New Mainstack, File menu > Move Substack to File..., About main stacks, substacks, and the organization of a stack file, Why does an unwanted handler run?, create stack command, mainStack property

On menu File

Creates a new untitled stack in the same file as the active main stack. When you save the substack, it is saved in the main stack's file. (This item is disabled if the active window is not a main stack.)

Open Stack...

See Also:

File menu > Open Recent Stack, File menu > Close, Tools menu > Application Browser, Window menu > Send Window to Back, How to respond when a stack opens, Why can't I open a downloaded stack?, Why does importing a HyperCard stack cause an error?, Why does my stack open slowly?, Why does Revolution ask to purge a stack?, Why don't the menus appear when I open a stack?, Why is there already a stack with the same name?, go command, topLevel command

On menu File

Opens the main stack whose file you select.

If you select a HyperCard file, it is automatically converted into a Revolution main stack.

Tip: To see all files, hold down the Option key (Mac OS or OS X) or Alt key (Windows or Unix) while choosing the menu item.

Open Recent Stack

See Also:

File menu > Open Stack..., Why can't I find a stack I just saved?, Why does Revolution ask to purge a stack?, go command, topLevel command

On menu File

Opens a cascading menu containing the names of the 25 main stacks you have most recently closed. Choose a name to open the stack.

Close

See Also:

File menu > Close and Remove from Memory..., File menu > Save, File menu > Revert to Saved..., How to close a window, How to respond to closing a window, close command

On menu File

Closes the active window. (This item is disabled if no window is open.)

Close and Remove from Memory...

See Also:

File menu > Close, File menu > Save, How to close a window, delete stack command, destroyStack property

On menu File

Closes the current stack and all stacks in the same stack file as the current stack, and removes all stacks in the file from memory. (This item is disabled if the active window is not a stack.)

Import as Control

See Also:

File menu > New Referenced Control, Object menu > New Control, create command, import command

On menu File

Opens a cascading menu you can use to choose a file and place its contents in a new control of the appropriate type. (This item is disabled if the active window is not a stack.)

Image File...

See Also:

File menu > Import as Control > Snapshot, File menu > Import as Control > All Images in Folder..., File menu > Import as Control > Image into Library..., Object menu > New Control > Image..., Image Types Reference, How to display a TIFF file, How to import a picture file into an existing image object, Why does an image disappear?, create command, import command

On menu File > Import as Control

Imports the picture file you choose as a new image on the current card. You can import GIF, JPEG, PNG, BMP, XWD, XBM, XPM, or PBM, PGM, or PBM files (and PICT files on Mac OS and OS X systems).

Snapshot

See Also:

File menu > Import as Control > Image File..., How to create a screenshot of a stack, import snapshot command

On menu File > Import as Control

Displays a crosshairs cursor for you to select an area of the screen, and imports a screen shot of that area as a new image on the current card.

Audio File...

See Also:

File menu > Import as Control > All Audio Files in Folder..., Object menu > New Control > Player, create command, import command

On menu File > Import as Control

Imports the sound file you choose as a new audio clip in the current stack. You can import WAV, AIFF, or AU files.

Video File...

See Also:

Object menu > New Control > Player, create command, import command

On menu File > Import as Control

Imports the video file you choose as a new video clip in the current stack. You can import QuickTime, AVI, or MPEG files.

Text File...

See Also:

How to import a text file into a field, Object menu > New Control > Field, create command, read from file command

On menu File > Import as Control

Imports the text file you choose as a new field on the current card.

EPS File...

See Also:

Why does an image disappear?, create command, import command

On menu File > Import as Control

Imports the Encapsulated PostScript file you choose as a new EPS object on the current card. (This item is disabled on Mac OS, OS X, and Windows systems.)

All Images in Folder...

See Also:

File menu > Import as Control > Image File..., File menu > Import as Control > All Images in Folder into Library..., File menu > New Referenced Control > All Images in Folder..., Object menu > New Control > Image, Why does an image disappear?, create command, files function, import command

On menu File > Import as Control

Imports all the picture files in the folder you choose, and places them in new images on the current card. (Subfolders, and other types of files, are ignored.) You can import GIF, JPEG, PNG, BMP, XWD, XBM, XPM, or PBM, PGM, and PBM files (and PICT files on Mac OS and OS X systems).

All Audio Files in Folder...

See Also:

Object menu > New Control > Player, create command, files function, import command

On menu File > Import as Control

Imports all the sound files in the folder you choose, and places them in new audio clips in the current stack. (Subfolders, and other types of files, are ignored.) You can import WAV, AIFF, and AU files.

New Referenced Control

See Also:

File menu > Import as Control, Object menu > New Control, create command, filename property

On menu File

Opens a cascading menu you can use to select a file to associate with a new control of the appropriate type. (This item is disabled if the active window is not a stack.)

Image File...

See Also:

File menu > Import as Control > Image File..., File menu > New Referenced Control > All Images in Folder..., Object menu > New Control > Image, Image Types Reference, How to display a picture from a web server, Why does an image disappear?, create command, filename property

On menu File > New Referenced Control

Creates a new image on the current card and displays the picture file you select in the new image object. You can reference GIF, JPEG, PNG, BMP, XWD, XBM, XPM, or PBM, PGM, or PBM files (and PICT files on Mac OS and OS X systems).

Quicktime-Supported File...

See Also:

File menu-> Import as Control -> Video File..., File menu -> Import as Control -> Audio file..., Object menu > New Control > Player, How to play a streaming QuickTime file, Why don't movies play?, create command, filename property

On menu File > New Referenced Control

Creates a new player on the current card and associates the audio or video file you select with the new player.

All Images in Folder...

See Also:

File menu > Import as Control > All Images in Folder..., File menu > New Referenced Control > Image File..., Object menu > New Control > Image, create command, filename property, files function

On menu File > New Referenced Control

For each picture file in the folder you select, creates a new image on the current card and associates it with one of the files. (Subfolders, and other types of files, are ignored.) You can reference GIF, JPEG, PNG, BMP, XWD, XBM, XPM, or PBM, PGM, or PBM files (and PICT files on Mac OS and OS X systems).

Save

See Also:

(Platform-dependent)

On menu File

Saves changes to the current stack and to any other stacks that reside in the same stack file. If the file has not yet been saved, you specify the new file's name and location. (This item is disabled if the active window is not a stack.)

Hold down the Option key (Mac OS or OS X systems), the Meta key (Unix systems), or the Alt key (Windows systems) to save all open stacks.

Save As...

See Also:

File menu > Save, File menu > Move Substack to File..., File menu > Build Distribution..., How to create a file, save command

On menu File

Saves the current stack, along with any other stacks that reside in the same file, to a new file with a name and location you specify. The new file becomes the current working copy. (This item is disabled if the active window is not a stack.)

Move Substack to File...

See Also:

File menu > New Substack of (main stack name), File menu > Save As..., Tools menu > Application Browser, mainStack property, save command

On menu File

Saves the frontmost substack as a main stack in a file of its own, with a name and location you specify. The substack is removed from its previous stack file. (This item is disabled if the active window is not a substack.)

Revert to Saved...

See Also:

File menu > Close, File menu > Close and Remove From Memory..., File menu > Save, revert command

On menu File

Throws away any changes to the current stack, along with any other stacks that reside in the same stack file. All stacks in the file revert to the last version saved. (This item is disabled if the active window is not a stack.)

Build Distribution...

See Also:

File menu > Save As..., Tools menu > Application Browser, Supported Platforms Reference, Why can't I create a standalone MacOS application?

On menu File

Packages the current stack for distribution as files or a standalone application. (This item is disabled if the active window is not a stack.)

Page Setup...

See Also:

File menu > Print Card..., File menu > Print Field..., How to display the print settings dialog box, answer printer command, revShowPrintDialog command

On menu File

Opens the Page Setup dialog box for the currently selected printer.

Print Card...

See Also:

File menu > Page Setup..., File menu > Print Field..., Tools menu > Report Builder, Recipe for printing all cards that contain a word, How to create a custom printed report, How to display the print settings dialog box, How to print a card, How to print all the cards in a stack, print command

On menu File

Prints the current card. (This item is disabled if the active window is not a stack.)

Print Field...

See Also:

File menu > Page Setup..., File menu > Print Card..., Tools menu > Report Builder, How to display the print settings dialog box, How to print the contents of a field, print command, revPrintField command, revPrintText command

On menu File

Prints the currently selected field. (This item is disabled if no field is selected or there is no text selection.)

Quit

See Also:

Why does my application quit when I close its windows?, quit command

On menu File

Closes all open stacks and quits Revolution.

Edit

See Also:

The Edit menu contains commands to select, cut, copy, and paste text and objects.

Undo

See Also:

undo command

On menu Edit

Reverses the most recent text change, paint action, or movement or deletion of an object.

Cut

See Also:

How to select a card, cut command

On menu Edit

Removes the selected text or object and places it on the clipboard. (This item is disabled if nothing is selected.)

The wording of the item is "Cut Text" if text is selected, "Cut Objects" if one or more objects is selected, and "Cut" if nothing is selected.

Copy

See Also:

How to select a card, copy command

On menu Edit

Places a copy of the selected text, image portion, or objects on the clipboard. (This item is disabled if nothing is selected.)

The wording of the item is "Copy Text" if text is selected, "Copy Objects" if one or more objects is selected, and "Copy" if nothing is selected.

Paste

See Also:

paste command

On menu Edit

Copies the contents of the clipboard to the current stack or text field. (This item is disabled if the clipboard does not contain text or objects.)

The wording of the item is "Paste Text" if the clipboard contains text, "Paste Objects" if the clipboard contains one or more objects, and "Paste" if the clipboard is empty or contains another kind of data.

Clear

See Also:

delete command

On menu Edit

Removes the selected text or objects, without placing it on the clipboard. (This item is disabled if nothing is selected.)

The wording of the item is "Clear Text" if text is selected, "Clear Objects" if one or more objects is selected, and "Clear" if nothing is selected.

Duplicate

See Also:

Edit menu > Replicate, How to duplicate an object, How to select a card, Shortcut to duplicate a control, clone command

On menu Edit

Makes a copy of the selected object or objects. If the object is a card, the copy is added after the current card. If the object is a control, the copy is placed on the current card, below and to the right of the original object. (This item is disabled if no object is selected.)

Replicate...

See Also:

Edit menu > Duplicate, How to duplicate an object, clone command

On menu Edit

Makes one or more copies of the selected object or objects, using the settings you select. (This item is disabled if no object is selected.)

Select All

See Also:

Shortcut to select multiple controls, select command, selection keyword

On menu Edit

Selects all the text in the current field (if there is a text selection or insertion point) or all the controls on the current card. (This item is disabled if the active window is not a stack and there is no text selection.)

Deselect All

See Also:

select command, selection keyword

On menu Edit

Deselects any selected objects, or removes the insertion point from a field. (This item is disabled if nothing is selected and there is no text insertion point.)

Invert Selection

See Also:

select command, selection keyword

On menu Edit

Selects all the unselected objects and unselects all the selected ones. (This item is disabled if no objects are selected.)

Select Grouped Controls

See Also:

About groups and backgrounds, selectGroupedControls property

On menu Edit

If this item is checked, clicking a control thatÆs part of a group selects only that control. If unchecked, clicking a control thatÆs part of a group selects the group.

Intersected Selections

See Also:

selectionMode property

On menu Edit

If this item is checked, dragging with the Pointer tool selects each object that intersects the dragged rectangle. If unchecked, dragging with the Pointer tool selects only objects that are entirely enclosed by the dragged rectangle.

Find and Replace...

See Also:

How to search a stack, Why does Revolution ask to purge a stack?, Recipe for a Find field, find command, globalNames function, selection keyword

On menu Edit

Searches for and optionally replaces text in fields, properties, scripts, global variables, or button contents.

Preferences

See Also:

How to disable tool tips, How to hide other applicationsÆ windows, backdrop property, destroyStack property, grid property, gridSize property, navigationArrows property, templateButton keyword, templateField keyword, templateGraphic keyword, templateImage keyword, templateScrollbar keyword, toolTipDelay property

On menu Edit

Sets application-wide preferences.

Tools

See Also:

The Tools menu contains commands to work with RevolutionÆs tool palettes and to use stack development tools.

Browse Tool

See Also:

Shortcut to switch between browse and pointer tool, browse keyword, choose command

On menu Tools

Chooses the Browse tool for performing user actions (such as clicking buttons or entering text).

Pointer Tool

See Also:

Shortcut to switch between browse and pointer tool, choose command, pointer keyword

On menu Tools

Chooses the Pointer tool for selecting, moving, and resizing objects.

Tools Palette

See Also:

Shortcut to hide Revolution palettes, choose command, tool function

On menu Tools

Shows or hides the Tools palette for choosing tools for stack creation.

Application Browser

See Also:

About main stacks, substacks, and the organization of a stack file

On menu Tools

Opens the Application Browser window, which lists all open stacks, the cards in each stack, and the controls on each card.

Animation Builder

See Also:

Animation Builder Tutorial, How to animate a sprite, `revGoToFramePaused` command, `revPlayAnimation` command, `revStopAnimation` command

On menu Tools

Creates a frame-based animation using the objects you specify.

Menu Builder

See Also:

About menus and the menu bar, Menu Builder Tutorial, How to switch between menu bars, Why does a menu item have missing characters?, Why does the top of my stack window disappear when I add a menu bar?, defaultMenubar property, menubar property

On menu Tools

Creates or changes a custom menu or menu bar.

Database Query Builder

See Also:

About connecting to and using SQL databases, `revOpenDatabase` function

On menu Tools

Creates settings to use with a SQL database. Using the Database Query Builder, you can connect to a database and specify a SQL query to generate a cursor result set.

Connection settings and cursor result sets can be linked to fields for display of the data, using the Database Linked pane in the field's property inspector.

Report Builder

See Also:

File menu > Print Card..., File menu > Print Field..., Object menu > New Control > Report Object, View menu > Go to Report Page, How to create a report layout card, How to create and store report printing settings for later use, How to print a report, revPrintReport command

On menu Tools

Creates settings for printed reports, and prints the report viewers on the current card.

Message Box

See Also:

How to determine which pending messages have been scheduled, message box keyword

On menu Tools

Shows or hides the message box.

Object

See Also:

The Object menu contains commands to change the properties of the selected object or objects, to create new objects, and to work with groups.

Object Inspector

See Also:

Object menu > Card Inspector, Object menu > Stack Inspector, Object menu > Object Script, About custom properties and custom property sets, About properties and property profiles, How to display an object's property inspector, How to show more than one object's properties at a time, Why doesn't a custom property appear in the property inspector?, Shortcut to change a control's properties, Shortcut to display a contextual menu, Shortcut to display a contextual menu (pointer tool), Shortcut to display a control's property inspector, properties property, set command

On menu Object

Sets properties for the selected objects. If more than one object is selected, changes made to the properties (except the object's script) are applied to each of the selected objects. (This item is disabled if no object is selected.)

Card Inspector

See Also:

Object menu > Object inspector, Object menu > Stack Inspector, Object menu > Card Script, About custom properties and custom property sets, About properties and property profiles, How to display an object's property inspector, How to select a card, How to show more than one object's properties at a time, Why doesn't a custom property appear in the property inspector?, properties property, set command

On menu Object

Sets properties for the current card. (This item is disabled if the active window is not a stack.)

Stack Inspector

See Also:

Object menu > Object Inspector, Object menu > Card Inspector, Object menu > Stack Script, About custom properties and custom property sets, About properties and property profiles, How to display an object's property inspector, How to show more than one object's properties at a time, Why doesn't a custom property appear in the property inspector?, properties property, set command

On menu Object

Sets properties for the current stack. (This item is disabled if the active window is not a stack.)

Object Script

See Also:

How to change a script at run time, Shortcut to edit a control's script, script property, set command

On menu Object

Opens the script editor for the selected objects. If more than one object is selected, a script editor window opens for each one. (This item is disabled if no object is selected.)

Card Script

See Also:

How to change a script at run time, Shortcut to edit the card script, script property

On menu Object

Opens the script editor for the current card. (This item is disabled if the active window is part of the Revolution development environment.)

Stack Script

See Also:

How to change a script at run time, Shortcut to edit the stack script, script property

On menu Object

Opens the script editor for the current stack. (This item is disabled if the active window is part of the Revolution development environment.)

Group Selected

See Also:

About groups and backgrounds, group command

On menu Object

Makes the selected objects into a group. This item changes to "Ungroup Selected" if the only selected object is a group. (This item is disabled if no object is selected.)

Ungroup Selected

See Also:

ungroup command

On menu Object

Makes the selected group into individual objects. This removes the group permanently if you visit another card before grouping the objects again. This item changes to "Group Selected" if more than one object is selected. (This item is disabled if no object is selected.)

Edit Group

See Also:

editBackground property, start editing command

On menu Object

Enters group-editing mode to make changes to the objects in the selected group. This item changes to **Stop Editing Group** while in group-editing mode. (This item is disabled if no object is selected, if more than one object is selected, or if the selected object is not a group.)

Stop Editing Group

See Also:

editBackground property, stop editing command

On menu Object

Leaves group-editing mode. This item changes to "Edit Group" if the stack is not already in group-editing mode. (This item is disabled if no object is selected, if more than one object is selected, or if the selected object is not a group.)

Remove Group

See Also:

remove command

On menu Object

Removes the selected group from the current card, without deleting it from the stack. (This item is disabled if no object is selected, if more than one object is selected, or if the selected object is not a group.)

Place Group

See Also:

About groups and backgrounds, How to automatically include groups on a new card, How to include a group on a card, How to list the groups in a stack, place command

On menu Object

Opens a cascading menu containing the names of groups that are in the stack but not on the current card. Choose a group to add it to the current card. (This item is disabled if all the groups in the stack already appear on the current card, or if there are no groups.)

Note: Only the names of top-level groups are listed; groups that are a part of another group are not listed in this menu.

New Card

See Also:

Object menu > Delete Card, How to create a card template for a stack, Why does the window change size when I create a card?, Recipe for a Cards menu, create card command, templateCard keyword

On menu Object

Creates a new card following the current card. (This item is disabled if the active window is not a stack.)

Note: If there are any groups on the current card when you choose this menu item, they are automatically placed on the new card. (If a group's `backgroundBehavior` is false, it is not placed automatically on new cards.)

Delete Card

See Also:

Object menu > New Card, delete command

On menu Object

Deletes the current card from the frontmost stack. (This item is disabled if the active window is not a stack.)

New Control

See Also:

create command

On menu Object

Opens a cascading menu you can use to create a new control. (This item is disabled if the active window is not a stack.)

Standard Button

See Also:

create command, standard keyword

On menu Object > New Control

Creates a standard click button (a button object whose style property is set to `standard`) on the current card.

Rectangle Button

See Also:

create command, rectangle keyword

On menu Object > New Control

Creates a rectangular button (a button object whose style property is set to `rectangle`) on the current card.

Shadow Button

See Also:

create command, shadow keyword

On menu Object > New Control

Creates a standard button with a drop shadow (a button object whose style property is set to `shadow`) on the current card.

Blank Button

See Also:

create command, opaque keyword

On menu Object > New Control

Creates an opaque, borderless button (a button object whose style property is set to `opaque`) on the current card.

Checkbox

See Also:

How to give a checkbox a different state on each card, create command

On menu Object > New Control

Creates a check box (a button object whose style property is set to `checkbox`) on the current card.

Radio Button

See Also:

How to change the selected button in a radio button cluster, How to create a radio button cluster, create command, style property

On menu Object > New Control

Creates a radio button (a button object whose style property is set to `radioButton`) on the current card.

Tabbed Button

See Also:

create command, menu keyword, tabbed keyword

On menu Object > New Control

Creates a tabbed menu button (a button object whose style property is set to `menu` and whose `menuMode` property is set to `tabbed`) on the current card.

Report Object

See Also:

Tools menu > Report Builder, View menu > Go to Report Page, How to create a report layout card, How to display the contents of a field in a printed report

On menu Object > New Control

Creates a new report viewer object on the current card.

Field

See Also:

create command, rectangle keyword

On menu Object > New Control

Creates a text field (a field object whose style property is set to `rectangle`) on the current card.

Scrolling Field

See Also:

create command

On menu Object > New Control

Creates a scrolling text field (a field object whose style property is set to `scrolling`) on the current card.

List Field

See Also:

create command, listBehavior property, rectangle keyword

On menu Object > New Control

Creates a clickable list (a field object whose style property is set to `rectangle` and whose listBehavior property is set to true) on the current card.

Scrolling List Field

See Also:

create command, listBehavior property

On menu Object > New Control

Creates a scrolling clickable list (a field object whose style property is set to `scrolling` and whose listBehavior property is set to true) on the current card.

Label Field

See Also:

create command, showBorder property, transparent keyword

On menu Object > New Control

Creates a transparent, borderless field that can be used as a text label (a field object whose style property is set to `transparent` and whose showBorder property is set to false).

Horizontal Scrollbar

See Also:

create command

On menu Object > New Control

Creates a scrollbar object oriented horizontally.

Vertical Scrollbar

See Also:

create command

On menu Object > New Control

Creates a scrollbar object oriented vertically.

Scale Bar

See Also:

create command

On menu Object > New Control

Creates a horizontal slider control (a scrollbar object whose style property is set to `scale`).

Progress Bar

See Also:

create command

On menu Object > New Control

Creates a blank horizontal progress bar (a scrollbar object whose style property is set to `progressbar`).

Pulldown Menu

See Also:

create command, pulldown keyword

On menu Object > New Control

Creates a button whose contents appear as a pulldown menu when the button is clicked (a button object whose style property is set to `menu` and whose `menuMode` property is set to `pulldown`).

Popup Menu

See Also:

create command, popup keyword

On menu Object > New Control

Creates a button whose contents appear as a popup menu when the button is clicked (a button object whose style property is set to `ômenuö` and whose `menuMode` property is set to `ôpopupö`).

Option Menu

See Also:

create command, option keyword

On menu Object > New Control

Creates a button whose contents appear as an option menu when the button is clicked (a button object whose style property is set to `menu` and whose `menuMode` property is set to `option`).

An option menu is displayed as a popup menu on Mac OS and OS X systems, an option menu on Unix systems, or a drop-down list on Windows systems.

Combo Box Menu

See Also:

comboBox keyword, create command

On menu Object > New Control

Creates a button whose contents appear as a combo box when the button is clicked (a button object whose style property is set to `menuItem` and whose `menuMode` property is set to `comboBox`).

Menu Item

See Also:

create command

On menu Object > New Control

Creates a button that can be used as a menu item in a stack menu (a button object whose style property is set to `opaque`, whose `autoArm` property is set to `true`, and whose text is left-aligned).

Cascade Menu Item

See Also:

`cascade` keyword, `create command`

On menu Object > New Control

Creates a button that can be used as a cascading menu in a stack menu (a button object whose style property is set to `menu`, whose `menuMode` property is set to `cascade` and , whose `autoArm` property is set to `true`).

Divider Button

See Also:

create command

On menu Object > New Control

Creates a horizontal dividing line used in menus (a narrow button object whose style property is set to `standard`) on the current card.

Rectangle Graphic

See Also:

create command, rectangle keyword

On menu Object > New Control

Creates a square or rectangle (a graphic object whose style property is set to `rectangle`).

Oval Graphic

See Also:

create command, oval keyword

On menu Object > New Control

Creates an oval or circle (a graphic object whose style property is set to oval).

Curve Graphic

See Also:

create command, curve keyword

On menu Object > New Control

Creates a curved line (a graphic object whose style property is set to `curve`).

Round Rect Graphic

See Also:

create command, roundRect keyword

On menu Object > New Control

Creates a square or rectangle with rounded corners (a graphic object whose style property is set to `roundRect`).

Polygon Graphic

See Also:

create command, polygon keyword

On menu Object > New Control

Creates an irregular polygon (a graphic object whose style property is set to `ôpolygonö`).

Line Graphic

See Also:

create command, line keyword

On menu Object > New Control

Creates a line (a graphic object whose style property is set to `line`).

Regular Polygon Graphic

See Also:

create command, regular keyword

On menu Object > New Control

Creates a regular polygon (a graphic object whose style property is set to `regular`). By default, the polygon is a four-sided diamond shape.

Image

See Also:

File menu > Import As Control > Image File..., File menu > New Referenced Control > Image File..., create command

On menu Object > New Control

Creates an empty image object. (You use the paint tools to paint inside an image.)

Player

See Also:

File menu > New Referenced Control > Quicktime-Supported File..., create command

On menu Object > New Control

Creates an empty player object for movie or sound files.

Flip

See Also:

On menu Object

Opens a cascading menu you can use to change the orientation of the selected image or graphic. (This item is disabled if any object other than an image or graphic is selected.)

Horizontal

See Also:

On menu Object > Flip

Swaps the left and right edges of the selected image or graphic, flipping it around an imaginary line drawn from top to bottom of the object.

Vertical

See Also:

On menu Object > Flip

Swaps the top and bottom edges of the selected image or graphic, flipping it around an imaginary line drawn from left to right of the object.

Rotate

See Also:

On menu Object

Opens a cascading menu you can use to rotate the selected image or graphic. (This item is disabled if any object other than an image or graphic is selected.)

By...

See Also:

On menu Object > Rotate

Rotates the selected image or graphic by the number of degrees you specify.

90 Right

See Also:

On menu Object > Rotate

Rotates the selected image or graphic 90 degrees to the right (clockwise).

90° Left

See Also:

On menu Object > Rotate

Rotates the selected image or graphic 90 degrees to the left (counterclockwise).

180 

See Also:

On menu Object > Rotate

Rotates the selected image or graphic 180 degrees (halfway around).

Align Selected Controls

See Also:

Shortcut to align control edges, bottom property, left property, rectangle property, right property, top property

On menu Object

Opens a cascading menu you can use to line up objects. (This item is disabled if no object or only one object is selected.)

Left

See Also:

Shortcut to align control edges, left property

On menu Object > Align Selected Controls

Moves the selected controls so their left edges are lined up with the left edge of the first control selected.

Right

See Also:

Shortcut to align control edges, right property

On menu Object > Align Selected Controls

Moves the selected controls so their right edges are lined up with the right edge of the first control selected.

Top

See Also:

Shortcut to align control edges, top property

On menu Object > Align Selected Controls

Moves the selected controls so their top edges are lined up with the top edge of the first control selected.

Bottom

See Also:

Shortcut to align control edges, bottom property

On menu Object > Align Selected Controls

Moves the selected controls so their bottom edges are lined up with the bottom edge of the first control selected.

Make Widths Equal

See Also:

Shortcut to equalize widths of selected controls, width property

On menu Object > Align Selected Controls

Resizes the selected controls so that the width of each one is equal to the width of the first control selected.

Make Heights Equal

See Also:

Shortcut to equalize widths of selected controls, height property

On menu Object > Align Selected Controls

Resizes the selected controls so that the height of each one is equal to the height of the first control selected.

Send to Back

See Also:

layer property

On menu Object

Moves the selected objects behind all other objects on the card. (This item is disabled if no object is selected.)

Move Backward

See Also:

layer property

On menu Object

Moves the selected objects back one layer. (This item is disabled if no object is selected.)

Move Forward

See Also:

layer property

On menu Object

Moves the selected objects forward one layer. (This item is disabled if no object is selected.)

Bring to Front

See Also:

How to bring a control to the front, layer property

On menu Object

Moves the selected objects in front of all other objects on the card. (This item is disabled if no object is selected.)

Text

See Also:

Recipe for automatically updating a Text menu, Recipe for handling selection of choices from a Text menu, Recipe for populating a Text menu, Shortcut to remove font changes from text

The Text menu contains commands to change the appearance of text.

Plain

See Also:

How to remove all styles from text, Shortcut to remove font changes from text, plain keyword, textStyle property

On menu Text

Removes all styles from the selected text or the selected object. (This item is disabled if nothing is selected.)

Bold

See Also:

bold keyword, textStyle property

On menu Text

Boldfaces the selected text, or all text in the selected object. (This item is disabled if nothing is selected.)

Italic

See Also:

italic keyword, textStyle property

On menu Text

Italicizes the selected text, or all text in the selected object. (This item is disabled if nothing is selected.)

Underline

See Also:

textStyle property, underline keyword

On menu Text

Underlines the selected text, or all text in the selected object. (This item is disabled if nothing is selected.)

Strikeout

See Also:

strikeout keyword, textStyle property

On menu Text

Draws a line through the selected text, or all text in the selected object. (This item is disabled if nothing is selected.)

Box

See Also:

box keyword, textStyle property

On menu Text

Draws a border around the selected text, or all text in the selected object. (This item is disabled if nothing is selected.)

3D Box

See Also:

textStyle property, threeDBox keyword

On menu Text

Draws a three-dimensional-look box around the selected text, or all text in the selected object. (This item is disabled if nothing is selected.)

Link

See Also:

How to create a hypertext link, How to remove the underline from linked text, Why is some text blue and underlined?, hide groups command, link keyword, show groups command, textStyle property

On menu Text

Makes the selected text, or all text in the selected object, a link. (This item is disabled if nothing is selected.)

Subscript

See Also:

Text menu > Superscript, How to make subscripts and superscripts, textShift property

On menu Text

Moves the selected text below the baseline and makes it smaller. (This item is disabled if no text is selected.)

Superscript

See Also:

Text menu > Subscript, How to make subscripts and superscripts, textShift property

On menu Text

Moves the selected text above the baseline and makes it smaller. (This item is disabled if no text is selected.)

Font

See Also:

How to change the font of text, Recipe for creating a font list for a menu, fontNames function, textFont property

On menu Text

Opens a cascading menu you can use to change the font face used for the selected text or objects. (This item is disabled if nothing is selected.)

Use Owner's Font

See Also:

Shortcut to remove font changes from text, effective keyword, textFont property

On menu Text > Font

Allows the font face of the object's owner (if an object is selected) or the object the text is in (if text is selected) to be used, removing any font face specific to the selected text or objects. (This item is disabled if nothing is selected.)

Size

See Also:

textSize property

On menu Text

Opens a cascading menu you can use to change the font size used for the selected text or objects. (This item is disabled if nothing is selected.)

Use Owner's Size

See Also:

Shortcut to remove font changes from text, effective keyword, textSize property

On menu Text > Size

Allows the font size of the object's owner (if an object is selected) or the object the text is in (if text is selected) to be used, removing any font size specific to the selected text or objects. (This item is disabled if nothing is selected.)

8

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text or objects to 8-point. (This item is disabled if nothing is selected.)

9

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text or objects to 9-point. (This item is disabled if nothing is selected.)

10

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text or objects to 10-point. (This item is disabled if nothing is selected.)

12

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text or objects to 12-point. (This item is disabled if nothing is selected.)

14

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text or objects to 14-point. (This item is disabled if nothing is selected.)

18

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text or objects to 18-point. (This item is disabled if nothing is selected.)

24

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text or objects to 24-point. (This item is disabled if nothing is selected.)

36

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text or objects to 36-point. (This item is disabled if nothing is selected.)

48

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text or objects to 48-point. (This item is disabled if nothing is selected.)

Other...

See Also:

textSize property

On menu Text > Size

Opens a dialog box you can use to change the font size used for the selected text or objects. (This item is disabled if nothing is selected.)

Color

See Also:

About colors and color references, `foregroundColor` property

On menu Text

Opens a cascading menu you can use to change the font color used for the selected text or objects. (This item is disabled if nothing is selected.)

Use Owner's Color

See Also:

Shortcut to remove font changes from text, effective keyword, foregroundColor property

On menu Text > Color

Allows the font color of the object's owner (if an object is selected) or the object the text is in (if text is selected) to be used, removing any font color specific to the selected text or objects. (This item is disabled if nothing is selected.)

Black

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text, or the text color of the selected objects, to black (0,0,0). (This item is disabled if nothing is selected.)

White

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text, or the text color of the selected objects, to white (255,255,255). (This item is disabled if nothing is selected.)

Red

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text, or the text color of the selected objects, to red (255,0,0). (This item is disabled if nothing is selected.)

Green

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text, or the text color of the selected objects, to green (0,255,0). (This item is disabled if nothing is selected.)

Blue

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text, or the text color of the selected objects, to blue (0,0,255). (This item is disabled if nothing is selected.)

Yellow

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text, or the text color of the selected objects, to yellow (255,255,0). (This item is disabled if nothing is selected.)

Pen Color

See Also:

About colors and color references, Color Names Reference, How to change the pen color, foregroundColor property, penColor property

On menu Text > Color

Changes the selected text, or the text color used in the selected objects, to the current pen color set by the penColor property. (This item is disabled if nothing is selected.)

Align

See Also:

How to change the alignment of text, `textAlign` property

On menu Text

Opens a cascading menu you can use to change the text alignment (justification) used for the text in the selected objects. (This item is disabled if nothing is selected.)

Left

See Also:

Text menu > Align > Center, Text menu > Align > Right, How to change the alignment of text, textAlign property

On menu Text > Align

Left-aligns text in the selected object or objects, lining up the left edge of each line of text parallel to the object's left edge. If text in a field is selected, the contents of the entire field is left-aligned.

Center

See Also:

Text menu > Align > Left, Text menu > Align > Right, How to change the alignment of text, `textAlign` property

On menu Text > Align

Centers text in the selected object or objects, lining up the middle of each line of text along a vertical line down the middle of the object. If text in a field is selected, the contents of the entire field is centered.

Right

See Also:

Text menu > Align > Left, Text menu > Align > Center, How to change the alignment of text, textAlign property

On menu Text > Align

Right-aligns text in the selected object or objects, lining up the right edge of each line of text parallel to the object's right edge. If text in a field is selected, the contents of the entire field is right-aligned.

Development

See Also:

The Development menu contains commands for debugging and for using custom tool stacks.

Object Library

See Also:

Why do icons disappear from a standalone application?

On menu Development

Displays the Object Library palette, which stores pre-scripted objects you can copy into the current stack for use.

Image Library

See Also:

[cursor property](#), [icon property](#), [patterns property](#)

On menu Development

Displays the Image Library palette, which displays images you can either reference or copy into the current stack for use.

You can use the Image Library to show all the icons and cursors that come with Revolution, all the images in the current stack, or image libraries that you create.

Plugins

See Also:

How to add custom utilities to the development environment, How to install a plugin

On menu Development

Opens a cascading menu you can use to open custom tool stacks stored in the Plugins folder.

Plugin Settings

See Also:

How to add custom utilities to the development environment

On menu Development > Plugins

Customizes which messages are handled by the custom tool stacks stored in the Plugins folder.

Script Debug Mode

See Also:

breakpoint command, `executionError` property

On menu Development

If this item is checked, the debugger is enabled: the debugger window appears when a breakpoint is encountered during script execution, and you can enter the debugger when an execution error occurs. If unchecked, the debugger is disabled.

Clear All Breakpoints

See Also:

Debug menu > Clear All Breakpoints (Script Editor), breakpoint command

On menu Development

Removes all breakpoints that you've used the script editor to mark, in all open stacks.

Note: This menu item does not affect breakpoints set with the breakpoint command.

Message Watcher

See Also:

Development menu > Variable Watcher, Debug menu > Message Watcher (Script Editor), How to monitor messages as they're sent

On menu Development

Opens the Message Watcher window, which you use to keep track of which messages, function calls, getProp calls, and setProp triggers have been sent to which objects.

Variable Watcher

See Also:

Development menu > Message Watcher, Debug menu > Variable Watcher (Script Editor), How to monitor the value of variables while debugging

On menu Development

Opens the variable watcher window, which you use to keep track of the value of variables during debugging. When the debugger is not open, the variable watcher shows the value of global variables.

Suppress Errors

See Also:

Shortcut to close the Error window, `lockErrorDialogs` property

On menu Development

Prevents display of the error window when Revolution encounters a script error.

Suppress Messages

See Also:

lockMessages property

On menu Development

Prevents system messages (such as openCard and closeCard) from being sent during normal navigation.

Suspend Development Tools

See Also:

View menu > Revolution UI Elements in Lists, View menu > Look and Feel, hide command

On menu Development

Hides Revolution's menus, palettes, and other parts of the development environment, so that you can preview how your application will look and behave as a standalone, outside the development environment.

To also suspend Revolution libraries, hold down the Shift key while choosing this menu item.

View

See Also:

The View menu contains commands to move around in the current stack, to display and navigate through the structure of open stacks, and to show or hide development tools.

Go First

See Also:

Shortcut to go to the previous card, go command

On menu View

Goes to the first card in the current stack.

Go Prev

See Also:

Shortcut to go to the previous card, go command

On menu View

Goes back to the previous card in the current stack.

Go Next

See Also:

Shortcut to go to the next card, go command

On menu View

Goes forward to the next card in the current stack.

Go Last

See Also:

Shortcut to go to the next card, go command

On menu View

Goes to the last card of the current stack.

Go Recent

See Also:

Shortcut to review the recent cards list, go command, recentCards property, recentNames property

On menu View

Goes back to the card you were on before navigating to the current card.

Go to Report Page

See Also:

Tools menu > Report Builder

On menu View

Opens a cascading menu you can use to preview each page of the report generated by the report viewers on the current card.

Note: If there are no report viewers on the current card, going to a report page has no effect.

First

See Also:

Tools menu > Report Builder, View menu > Go First

On menu View > Go to Report Page

Displays the content that will appear on the first page of the report, in the report viewers on the current card.

Note: If there are no report viewers on the current card, going to a report page has no effect.

Prev

See Also:

Tools menu > Report Builder, View menu > Go Prev

On menu View > Go to Report Page

Displays the content that will appear on the previous page of the report, in the report viewers on the current card.

Note: If there are no report viewers on the current card, going to a report page has no effect.

Next

See Also:

Tools menu > Report Builder, View menu > Go Next

On menu View > Go to Report Page

Displays the content that will appear on the next page of the report, in the report viewers on the current card.

Note: If there are no report viewers on the current card, going to a report page has no effect.

Last

See Also:

Tools menu > Report Builder, View menu > Go Last

On menu View > Go to Report Page

Displays the content that will appear on the last page of the report, in the report viewers on the current card.

Note: If there are no report viewers on the current card, going to a report page has no effect.

Number...

See Also:

Tools menu > Report Builder

On menu View > Go to Report Page

Displays the content that will appear on the specified page of the report, in the report viewers on the current card.

Note: If there are no report viewers on the current card, going to a report page has no effect.

Toolbar Text

See Also:

View menu > Toolbar Icons, hide command, show command

On menu View

Shows or hides the text labels in the Toolbar at the top of the screen. (To hide the Toolbar completely, uncheck both this item and ☐Toolbar Icons.)

Toolbar Icons

See Also:

View menu > Toolbar Text, hide command, show command

On menu View

Shows or hides the icons in the Toolbar at the top of the screen. (To hide the Toolbar completely, uncheck both this item and `ôToolbar Textö`.)

Palettes

See Also:

Shortcut to hide Revolution palettes, activatePalettes property, hide command, raisePalettes property, show command

On menu View

Shows or hides all open Revolution palettes.

Rulers

See Also:

hide command, show command

On menu View

Shows or hides a ruler at the left and bottom edges of each open stack.

Grid

See Also:

grid property

On menu View

If this item is checked, dragging and resizing objects is constrained by a pixel grid. If unchecked, you can drag and resize objects to any location.

Backdrop

See Also:

How to hide other applicationsÆ windows, backdrop property

On menu View

Shows or hides a solid or patterned backdrop behind RevolutionÆs windows.

Revolution UI Elements in Lists

See Also:

Tools menu > Application Browser, Development menu > Suspend Development Tools

On menu View

If this item is checked, elements of the Revolution development environment appear in lists: for example, development environment stacks appear in the Application Browser, and Revolution custom properties appear in the Custom Properties pane of the property inspector. If unchecked, elements of the Revolution development environment do not appear in such lists.

Look and Feel

See Also:

Supported Platforms Reference, lookAndFeel property

On menu View

Opens a cascading menu you can use to change the appearance of controls in order to preview your application's appearance on other platforms.

Mac OS Appearance Manager

See Also:

Supported Platforms Reference, lookAndFeel property

On menu View > Look and Feel

Returns to the usual appearance of stacks on Mac OS and OS X systems, using the Appearance Manager settings.

Mac OS Emulated

See Also:

Supported Platforms Reference, lookAndFeel property

On menu View > Look and Feel

Returns to the usual appearance of stacks on Mac OS systems, using the built-in drawing routines to emulate the standard Platinum look and feel.

Native Windows

See Also:

Supported Platforms Reference, lookAndFeel property

On menu View > Look and Feel

Returns to the usual appearance of stacks on Windows systems.

Native Linux

See Also:

Supported Platforms Reference, lookAndFeel property

On menu View > Look and Feel

Returns to the usual appearance of stacks on Unix systems.

Preview Mac OS

See Also:

Supported Platforms Reference, lookAndFeel property

On menu View > Look and Feel

Displays stacks as they will appear on Mac OS systems, allowing you to preview cross-platform applications while developing on a Unix or Windows system.

Preview Windows

See Also:

Supported Platforms Reference, lookAndFeel property

On menu View > Look and Feel

Displays stacks as they will appear on Windows systems, allowing you to preview cross-platform applications while developing on a Mac OS , OS X, or Unix system.

Preview Linux

See Also:

Supported Platforms Reference, lookAndFeel property

On menu View > Look and Feel

Displays stacks as they will appear on Unix systems, allowing you to preview cross-platform applications while developing on a Mac OS, OS X, or Windows system.

Show Invisible Objects

See Also:

showInvisibles property

On menu View

If this item is checked, objects whose visible property is set to false are shown. If unchecked, objects whose visible property is set to false remain hidden.

Window

See Also:

How to bring a window to the front, Recipe for a Cards menu, Recipe for a Window menu, openStacks function

The Window menu contains the names of open stack windows.

Send Window To Back

See Also:

layer property

On menu Window

Shuffles the frontmost stack window to the back and brings the second window to the front. (This item is disabled if only one stack is open or the active window is not a stack.)

Help

See Also:

The Help menu contains commands to find out more about using Revolution, license your copy of Revolution, get technical support via email, and check the Revolution site for software updates.

Documentation

See Also:

Help menu > Transcript Dictionary, How to quickly display the Transcript Dictionary, Shortcut to open Revolution documentation, help message

On menu Help

Opens the main Revolution documentation window, where you can get information about all aspects of developing in Revolution.

Transcript Dictionary

See Also:

Help menu > Documentation, How to quickly display the Transcript Dictionary

On menu Help

Opens the Transcript Dictionary, which explains all Transcript commands, functions, properties, control structures, messages, and keywords.

(The Transcript Dictionary is also available from the main Revolution Documentation window.)

Search Documentation...

See Also:

Help menu > Documentation

On menu Help

Opens a Search window that searches for a word or phrase in all sections of the documentation.

Tip: To see a list of topics where the search term appears, uncheck the ☐ See Search Term in Context box. To see the line where the phrase appears, as well as the topic name, check the box.

License Revolution...

See Also:

Help menu > Revolution Support, About upgrading Revolution from a previous version, scriptLimits function

On menu Help

Displays information about licensing your copy of Revolution, creates an order form, and accepts your license code to unlock your copy of Revolution.

Revolution Support

See Also:

How to report a bug or request a feature, How to request technical support, Help > Revolution Licensing, Help > Check for Revolution Updates

On menu Help

Displays the Revolution Support window, which you can use to send feedback or request technical support.

Check for Updates

See Also:

About upgrading Revolution from a previous version

On menu Help

Checks the Revolution web site for software updates that are available for download.

Script Editor Menu Reference

See Also:

edit command, script property

The Script Editor menu bar contains special commands to help you edit scripts. Click a menu name:

File menu (Script Editor)

Edit menu (Script Editor)

Text menu (Script Editor)

Script menu (Script Editor)

View menu (Script Editor)

Handler menu (Script Editor)

Debug menu (Script Editor)

Bookmarks menu (Script Editor)

Window menu (Script Editor)

File

See Also:

The File menu contains commands to print a script or close the script editor, with or without saving changes to the script.

Apply Script

See Also:

File menu > Apply Script and Close (Script Editor), File menu > Apply Script and Save Stack (Script Editor), Shortcut to close the script editor, script property

On menu File

Applies changes to the script that's currently being edited. (An applied script is not saved until the stack that contains it is saved.)

Close Script Editor

See Also:

File menu > Apply Script and Close (Script Editor), File menu > Quit (Script Editor), Shortcut to close the script editor

On menu File

Closes the script editor window.

Apply Script and Close

See Also:

File menu > Apply Script (Script Editor), File menu > Close Script Editor (Script Editor), Shortcut to close the script editor, script property

On menu File

Applies changes to the script that's currently being edited, and closes the script editor. (An applied script is not saved until the stack that contains it is saved.)

Apply Script and Save Stack

See Also:

File menu > Save, File menu > Apply Script (Script Editor), save command, script property

On menu File

Applies changes to the script that's currently being edited, and saves the stack file that the script belongs to. This saves the changes you've made to the script, along with any other changes made to any stack in the stack file.

Page Setup...

See Also:

File menu > Print Script... (Script Editor), answer printer command

On menu File

Opens the Page Setup dialog box for the currently selected printer.

Print Script...

See Also:

File menu > Print Field..., File menu > Page Setup... (Script Editor), print command

On menu File

Prints the script that's currently being edited.

Quit

See Also:

quit command

On menu File

Closes all open stacks and quits Revolution.

Edit

See Also:

The Edit menu contains commands to select, cut, copy, and paste text.

Undo

See Also:

undo command

On menu Edit

Reverses the most recent text change.

Cut

See Also:

cut command

On menu Edit

Removes the selected text and places it on the clipboard. (This item is disabled if nothing is selected.)

The wording of the item is ⌘Cut Text⌘ if text is selected and ⌘Cut⌘ if nothing is selected.

Copy

See Also:

copy command

On menu Edit

Places a copy of the selected text on the clipboard. (This item is disabled if nothing is selected.)

The wording of the item is "Copy Text" if text is selected and "Copy" if nothing is selected.

Paste

See Also:

paste command

On menu Edit

Copies the contents of the clipboard to the current script. (If the clipboard contains an object, the object is pasted into the current stack.)

The wording of the item is "Paste Text" if the clipboard contains text, "Paste Objects" if the clipboard contains an object, and "Paste" if the clipboard is empty or contains another kind of data.

Paste as Formatted String

See Also:

& operator, && operator, quote constant

On menu Edit

Places quotes around the contents of the clipboard and copies the resulting string to the current script. (If the text on the clipboard already contains quotes, or if there is more than one line, Revolution uses the quote and return constants and the & operator to create a string that can be included in a script without causing a script error.)

This item is disabled if the clipboard does not contain text.

Paste as Comment

See Also:

Script menu > Comment (Script Editor), Script menu > Uncomment (Script Editor), View menu > Default Comment Character... (Script Editor), -- keyword

On menu Edit

Copies the contents of the clipboard to the current script, with a comment character inserted before each line. (This item is disabled if the clipboard does not contain text.)

Clear

See Also:

delete command

On menu Edit

Removes the selected text, without placing it on the clipboard. (This item is disabled if nothing is selected.)

The wording of the item is "Clear Text" if text is selected and "Clear" if nothing is selected.

Select All

See Also:

select command, selection keyword

On menu Edit

Selects all the text in the script.

Deselect All

See Also:

select command, selection keyword

On menu Edit

Un-selects all the text in the script.

Text

See Also:

Shortcut to remove font changes from text

The Text menu contains commands to change the appearance of text.

Plain

See Also:

Shortcut to remove font changes from text, textStyle property

On menu Text

Removes all styles from the selected text. (This item is disabled if nothing is selected.)

Bold

See Also:

bold keyword, textStyle property

On menu Text

Boldfaces the selected text. (This item is disabled if nothing is selected.)

Italic

See Also:

italic keyword, textStyle property

On menu Text

Italicizes the selected text. (This item is disabled if nothing is selected.)

Underline

See Also:

textStyle property, underline keyword

On menu Text

Underlines the selected text. (This item is disabled if nothing is selected.)

Strikeout

See Also:

strikeout keyword, textStyle property

On menu Text

Draws a line through the selected text. (This item is disabled if nothing is selected.)

Box

See Also:

box keyword, textStyle property

On menu Text

Draws a border around the selected text. (This item is disabled if nothing is selected.)

3D Box

See Also:

textStyle property, threeDBox keyword

On menu Text

Draws a three-dimensional-look box around the selected text. (This item is disabled if nothing is selected.)

Link

See Also:

hide groups command, show groups command, textStyle property

On menu Text

Makes the selected text into a clickable link. (This item is disabled if nothing is selected.)

Subscript

See Also:

textShift property

On menu Text

Moves the selected text below the baseline. (This item is disabled if nothing is selected.)

Superscript

See Also:

textShift property

On menu Text

Moves the selected text above the baseline. (This item is disabled if nothing is selected.)

Font

See Also:

textFont property

On menu Text

Opens a cascading menu you can use to change the font face used for the selected text. (This item is disabled if nothing is selected.)

Use Default Font

See Also:

effective keyword, textSize property

On menu Text > Font

Sets the selected text to the default font chosen in the Default Fonts dialog box, removing any font specific to the selected text. (This item is disabled if nothing is selected.)

Size

See Also:

textSize property

On menu Text

Opens a cascading menu you can use to change the font size used for the selected text. (This item is disabled if nothing is selected.)

Use Default Size

See Also:

effective keyword, textSize property

On menu Text > Size

Sets the selected text to the default font size chosen in the Default Fonts dialog box, removing any font size specific to the selected text. (This item is disabled if nothing is selected.)

8

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text to 8-point. (This item is disabled if nothing is selected.)

9

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text to 9-point. (This item is disabled if nothing is selected.)

10

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text to 10-point. (This item is disabled if nothing is selected.)

12

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text to 12-point. (This item is disabled if nothing is selected.)

14

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text to 14-point. (This item is disabled if nothing is selected.)

18

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text to 18-point. (This item is disabled if nothing is selected.)

24

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text to 24-point. (This item is disabled if nothing is selected.)

36

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text to 36-point. (This item is disabled if nothing is selected.)

48

See Also:

textSize property

On menu Text > Size

Sets the font size of the selected text to 48-point. (This item is disabled if nothing is selected.)

Other...

See Also:

textSize property

On menu Text > Size

Opens a dialog box you can use to change the font size used for the selected text. (This item is disabled if nothing is selected.)

Color

See Also:

foregroundColor property

On menu Text

Opens a cascading menu you can use to change the font color used for the selected text. (This item is disabled if nothing is selected.)

Use Default Color

See Also:

effective keyword, textSize property

On menu Text > Color

Sets the selected text to the default font color of black, removing any color specific to the selected text.
(This item is disabled if nothing is selected.)

Black

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text to black (0,0,0). (This item is disabled if nothing is selected.)

White

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text to white (255,255,255). (This item is disabled if nothing is selected.)

Red

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text to red (255,0,0). (This item is disabled if nothing is selected.)

Green

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text to green (0,255,0). (This item is disabled if nothing is selected.)

Blue

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text to blue (0,0,255). (This item is disabled if nothing is selected.)

Yellow

See Also:

foregroundColor property

On menu Text > Color

Sets the color of the selected text to yellow (255,255,0). (This item is disabled if nothing is selected.)

Pen Color

See Also:

foregroundColor property, penColor property

On menu Text > Color

Changes the selected text to the current pen color set by the penColor property. (This item is disabled if nothing is selected.)

Align

See Also:

textAlign property

On menu Text

Opens a cascading menu you can use to change the text alignment (justification) used for the text in the selected objects. (This item is disabled if nothing is selected.)

Left

See Also:

Text menu > Align > Center, Text menu > Align > Right, textAlign property

On menu Text > Align

Left-aligns text in the script editor, lining up the left edge of each line of text parallel to the script editor's left edge.

Center

See Also:

Text menu > Align > Left, Text menu > Align > Right, textAlign property

On menu Text > Align

Centers text in the script editor, lining up the middle of each line of text along a vertical line down the middle of the script editor window.

Right

See Also:

Text menu > Align > Left, Text menu > Align > Center, textAlign property

On menu Text > Align

Right-aligns text in the script editor, lining up the right edge of each line of text parallel to the script editor's right edge.

Script

See Also:

The Script menu contains commands to comment or uncomment part of a script, format, colorize, or add the skeleton of a control structure.

Comment

See Also:

Script menu > Uncomment (Script Editor), View menu > Default Comment Character... (Script Editor),
-- keyword

On menu Script

Places a comment character at the beginning of the selected text. If more than one line is selected, a comment character is placed at the beginning of each line.

Uncomment

See Also:

Script menu > Comment (Script Editor), View menu > Default Comment Character... (Script Editor), -- keyword

On menu Script

Removes comment characters from the selected text.

Insert Control Structure

See Also:

On menu Script

Removes comment characters from the selected text.

If Then Else

See Also:

Script menu > Insert Control Structure > Switch, else keyword, end if keyword, if control structure, then keyword

On menu Script > Insert Control Structure

Inserts the skeleton of an if/then/else/end if control structure, on the line the insertion point is on.

Repeat

See Also:

repeat control structure

On menu Script > Insert Control Structure

Inserts the skeleton of a repeat control structure, on the line the insertion point is on.

Switch

See Also:

Script menu > Insert Control Structure > If Then Else, switch control structure

On menu Script > Insert Control Structure

Inserts the skeleton of a switch control structure, on the line the insertion point is on.

Try

See Also:

try control structure

On menu Script > Insert Control Structure

Inserts the skeleton of a try control structure, on the line the insertion point is on.

Colorize

See Also:

Script menu > Format (Script Editor), View menu > Live Colorization (Script Editor), View menu > Colorization Settings (Script Editor), foregroundColor property

On menu Script

Color-codes commands, functions, control structures, and other Transcript terms in the script to make it easier to read.

Format

See Also:

How to pretty print a script, Shortcut to format the current handler

On menu Script

Indents the script to show its structure.

View

See Also:

The View menu contains commands to format the script and change its appearance.

Look Up Selection In Transcript Dictionary

See Also:

Help menu > Documentation, Shortcut to look up Transcript term

On menu View

Opens the Transcript Dictionary as a mini-palette with basic information about the selected Transcript term. (Click the Expand button at the upper right to expand the palette to the full Transcript Dictionary window.)

Single-Handler View

See Also:

View menu > Show Handler List (Script Editor), script property

On menu View

If this item is checked, the script editor shows the contents of a single handler. If unchecked, the script editor shows the entire script.

Show Handler List

See Also:

View menu > Sort Handlers Alphabetically (Script Editor)

On menu View

If this item is checked, a clickable list of handlers in the current script is displayed on the left side of the script editor. If unchecked, the list is not displayed.

Sort Handler List Alphabetically

See Also:

View menu > Show Handler List (Script Editor)

On menu View

If this item is checked, the list displayed by the "Show Handler List" menu item, and the list in the Handlers menu, are sorted alphabetically. If unchecked, the lists appear in the same order as the handlers appear in the script.

Find and Replace

See Also:

Help menu > Search Documentation..., Term menu > Case Sensitive Searching (Script Editor), Term menu > Whole Word Searching (Script Editor), Shortcut to find the selection in scripts

On menu View

Chooses the Find and Replace utility to be displayed at the bottom of the script editor. Use the Find and Replace utility to search for text in the script and optionally replace it with other text.

Go to Line

See Also:

On menu View

Chooses the Go to Line utility to be displayed at the bottom of the script editor. Use the Go to Line utility to move the insertion point to a specified line number. If you enter two numbers X and Y separated by a comma, Go to Line goes to character Y of line X of the script.

Autocomplete

See Also:

Shortcut to complete words in scripts

On menu View

Chooses the Autocomplete utility to be displayed at the bottom of the script editor. Use the Autocomplete utility to have the script editor suggest words for auto-completion.

As you start to type a word, Transcript words that match what you've already entered appear at the bottom of the script editor. To choose one of the suggested words, press the Command key (Mac OS or OS X) or Control key (Unix or Windows) along with the number that appears before the suggested word.

Case Sensitive Searching

See Also:

Script menu > Find and Replace (Script Editor), caseSensitive property, find command

On menu View

If this item is checked, searches performed with the Find and Replace utility in the script editor are case-sensitive. If unchecked, searches are case-insensitive.

Whole Word Searching

See Also:

Script menu > Find and Replace (Script Editor), find command, whole keyword

On menu View

If this item is checked, searches performed with the Find and Replace utility in the script editor locate only complete words. If unchecked, searches locate either complete or partial words.

Live Colorization

See Also:

View menu > Colorization Settings... (Script Editor), foregroundColor property

On menu View

If this item is checked, the script editor automatically color-codes commands, control structures, and other Transcript terms as you type. If unchecked, no automatic color-coding is performed.

Colorization Settings...

See Also:

Script menu > Colorize (Script Editor), View menu > Live Colorization (Script Editor), `commandNames` function, `functionNames` function, `propertyNames` function

On menu View

Displays a dialog box containing settings for colorizing scripts.

Wrap Long Script Lines

See Also:

How to break a line in a script, keyword, dontWrap property

On menu View

If this item is checked, script lines that are too long to fit in the script editor without scrolling are wrapped around to the next line. If unchecked, long lines are not wrapped.

Default Fonts...

See Also:

Text menu > Font > Use Default Font (Script Editor), Text menu > Font > Use Default Size (Script Editor), Text menu > Font > Use Default Color (Script Editor), scriptTextFont property, scriptTextSize property

On menu View

Chooses the font, size, text height, and background color to use for scripts.

Default Comment Character...

See Also:

Script menu > Comment (Script Editor), Script menu > Uncomment (Script Editor), -- keyword

On menu View

Selects either # or -- as the comment marker to use for the Comment menu item. (The Uncomment menu item recognizes both synonyms.)

Handler

See Also:

View menu > Sort Handler List Alphabetically (Script Editor)

The Handler menu contains a list of all handlers in the current script. Choose a handler to go to that handler in the script.

Debug

See Also:

The Debug menu contains commands to aid in debugging scripts and to control when the debugger window appears during execution.

Set Breakpoint

See Also:

Debug menu > Clear All Breakpoints (Script Editor), breakpoint command

On menu Debug

Sets a temporary breakpoint at the current line of the script. (If more than one line is selected, the breakpoint is set at the first line.)

When a breakpoint is set, a marker appears in the left margin of the script.

Clear All Breakpoints

See Also:

Development menu > Clear All Breakpoints, Debug menu > Set Breakpoint (Script Editor)

On menu Debug

Removes all breakpoints in the current script.

Note: This menu item does not affect breakpoints set with the breakpoint command.

Step Into

See Also:

Debug menu > Step Over (Script Editor), Shortcut to step into the next line of a handler

On menu Debug

Executes the next line of the handler being debugged. If the next line is a call to another handler, this command steps into that handler.

(This menu item is disabled if the debugger is not running.)

Tip: Press Space to step into the next line.

Step Over

See Also:

Debug menu > Step Into (Script Editor), Shortcut to step over the next line of a handler

On menu Debug

Executes the next line of the handler being debugged. If the next line is a call to another handler, this command steps over that call, skipping it and staying within the current handler.

(This menu item is disabled if the debugger is not running.)

Tip: Press Option-Space (Mac OS or OS X) or Alt-Space (Unix or Windows) to step over the next line.

Trace

See Also:

Debug menu > Step Into (Script Editor), Debug menu > Step Over (Script Editor)

On menu Debug

Begins slowly executing the handler being debugged, starting from the current line.

(This menu item is disabled if the debugger is not running.)

Tip: The delay between lines can be changed by changing the value of the traceDelay property.

Run

See Also:

Debug menu > Abort (Script Editor)

On menu Debug

Resumes normally executing the handler being debugged, starting from the current line.

(This menu item is disabled if the debugger is not running.)

Abort

See Also:

Debug menu > Run (Script Editor)

On menu Debug

Stops running the handler being debugged.

(This menu item is disabled if the debugger is not running.)

Message Watcher

See Also:

Development menu > Message Watcher, Debug menu > Variable Watcher (Script Editor), How to monitor messages as they're sent

On menu Debug

Opens the Message Watcher window, which you use to keep track of which messages, function calls, getProp calls, and setProp triggers have been sent to which objects.

Variable Watcher

See Also:

Development menu > Variable Watcher, Debug menu > Message Watcher (Script Editor), How to monitor the value of variables while debugging

On menu Debug

Opens the variable watcher window, which you use to keep track of the value of variables during debugging. When the debugger is not open, the variable watcher shows the value of global variables.

Script Debug Mode

See Also:

Development menu > Script Debug Mode

On menu Debug

If this item is checked, the debugger is enabled: the debugger window appears when a breakpoint is encountered during script execution, and you can enter the debugger when an execution error occurs. If unchecked, the debugger is disabled.

Bookmarks

See Also:

The Bookmark menu contains a list of bookmark locations in the current script. Choose a bookmark to scroll to that location.

Add Bookmark...

See Also:

On menu Bookmarks

Stores the location of the selection or insertion point in the Bookmarks menu, under a name you specify. A special comment is added to the script at the bookmarked location.

Window

See Also:

The Window menu contains the names of open script editor windows.

Distribution Builder Menu Reference

See Also:

The Distribution Builder menu bar contains special commands to create, save, and use Distribution Builder settings files. Click a menu name:

File menu (Distribution Builder)

Edit menu (Distribution Builder)

File

See Also:

The File menu contains commands to open and save lists of settings for the creation of standalone applications and file distributions.

Open Distribution...

See Also:

File menu > Close (Distribution Builder)

On menu File

Opens the Distribution Builder settings file whose name you select. The settings are loaded into the Distribution Builder window, ready to be used.

Close

See Also:

close command

On menu File

Closes the Distribution Builder window.

Save

See Also:

File menu > Save Distribution As... (Distribution Builder), save command

On menu File

Saves changes to the Distribution Builder settings file. If you have specified settings, but not saved them in a file previously, you specify the new file's name and location.

Save Distribution As...

See Also:

File menu > Save (Distribution Builder), save command

On menu File

Saves the Distribution Builder settings to a new file with a name and location you specify. The new file becomes the current working copy.

Revert to Saved...

See Also:

File menu > Save (Distribution Builder), File menu > Save Distribution As... (Distribution Builder), File menu > Revert to Default Settings... (Distribution Builder), revert command

On menu File

Throws away any changes to the Distribution Builder settings.

Revert to Default Settings...

See Also:

File menu > Revert to Saved... (Distribution Builder)

On menu File

Restores the Distribution Builder's default settings, removing any changes you've made.

Build Distribution

See Also:

Why can't I create a standalone MacOS application?, Why doesn't the ask or answer dialog appear in my standalone?, save command

On menu File

Creates a standalone application or set of stack files for distribution, with the settings specified in the Distribution Builder window.

Quit

See Also:

quit command

On menu File

Closes all open stacks and quits Revolution.

Edit

See Also:

The Edit menu contains commands to cut, copy, and paste text.

Undo

See Also:

undo command

On menu Edit

Reverses the most recent text change.

Cut

See Also:

How to select a card, cut command

On menu Edit

Removes the selected text and places it on the clipboard. (This item is disabled if nothing is selected.)

Copy

See Also:

How to select a card, copy command

On menu Edit

Places a copy of the selected text on the clipboard. (This item is disabled if nothing is selected.)

Paste

See Also:

paste command

On menu Edit

Copies the contents of the clipboard to the current text insertion point or selection. (This item is disabled if the clipboard does not contain text.)

Clear

See Also:

delete command

On menu Edit

Removes the selected text, without placing it on the clipboard. (This item is disabled if nothing is selected.)

absolute coordinates

See Also:

Measurement of a position by its distance from the top and left edges of the screen.

absolute file path

See Also:

The full name and location of a file, beginning with the disk the file is on, including the names of all folders that contain the file in nested order, and finally the file name itself.

An absolute file path is always the same, regardless of which folder is the current default folder.

active control

See Also:

The control that receives any characters the user types. Typically, this control is the field with the current insertion point, but other types of control can also be the active control.

When a control becomes active, it is said to receive the focus.

active window

See Also:

The window that is frontmost on the screen. The active window receives clicks and typed characters, and its appearance may be different from the appearance of inactive windows.

AIFF

See Also:

Audio Interchange File Format. A format for sound files developed by Apple Computer and often used on Mac OS systems.

alias

See Also:

On Mac OS and OS X systems, a special file type that refers to another file, a folder, or a disk. The alias is an alternative icon for the original item, and double-clicking the alias opens the item it refers to.

(The Unix equivalent is the symbolic link. The Windows equivalent is the shortcut.)

alpha channel

See Also:

A set of values, or channel, that encodes transparency information for an image.

Each pixel in the image has a number that specifies how transparent it is. An image's alpha channel consists of the transparency numbers for all the pixels in the image.

alphanumeric character

See Also:

A character that is either a letter (A-Z or a-z) or a digit (0-9).

Punctuation marks and whitespace characters are not alphanumeric characters.

Alt key

See Also:

A modifier key on Windows systems, usually located at the right of the Control key.

(The Mac OS and OS X equivalent is the Option key. The Unix equivalent is the Meta key.)

animated GIF

See Also:

A GIF-format image that contains multiple pictures, each of which is one frame in an animation.

Animation library

See Also:

The Revolution custom library that supports animations created with the Animation Builder. The `revGoToFramePaused`, `revPlayAnimation`, and `revStopAnimation` commands are part of the Animation library.

anomaly

See Also:

An unexpected result; an inconsistency in a script or program. (The unsympathetic might refer to an anomaly as a ôbugö.)

antialiasing

See Also:

Blurring a sharp edge in an image, by using intermediate colors on the edge between the image and its background. This fools the eye into seeing a smooth edge instead of the "jaggies" that result from a diagonal or curved edge when seen on screen.

appearance

See Also:

The way controls, menus, windows, and so forth look on the screen. Part of the look and feel of a user interface, along with behavior.

append

See Also:

To add data to the end of a container or file, without removing its current contents.

Apple Event

See Also:

A protocol for interapplication communication on Mac OS and OS X systems.

An application can send an Apple Event to another application to make it do something such as print, open a file, run a script, etc.

Apple menu

See Also:

The leftmost menu in the menu bar on Mac OS and OS X systems.

On Mac OS systems, the Apple menu contains the current application's About menu item.

AppleScript

See Also:

System-wide scripting language developed by Apple Computer which can be used to control the computer system, launch programs, manipulate files and folders, and other tasks.

AppleScript is included in Mac OS and OS X.

AppleTalk

See Also:

Network protocol for local area networks developed by Apple Computer.

Application Browser

See Also:

Part of the Revolution development environment. In the Application Browser, you can see a list of all open stacks, and all objects in your stacks—hidden or visible—and change their properties.

You open the Application Browser by choosing Tools menu>Application Browser.

application bundle

See Also:

On OS X systems, a special type of folder that is used to store files that make up an application. The names of application bundles end with the extension `.app`.

An application bundle is always presented to the user as a single file, and is a single file as far as the user's experience is concerned, but is internally treated by the operating system as a folder.

Application menu

See Also:

On OS X systems, the menu that is second from the left—between the Apple menu and your application's menus.

The Application menu has the same name as the currently running application, and contains general commands such as "Quit application name".

application

See Also:

The program thatÆs currently running: either the Revolution development environment, or a standalone application created with Revolution.

Aqua

See Also:

The clickable look and feel designed for OS X. When the lookAndFeel property is set to Appearance Manager, a Revolution application displays the Aqua appearance when running on OS X.

argument

See Also:

A value that is passed to an operator, function or command.

arm

See Also:

To change the appearance of a control (such as a button) when the mouse pointer moves into it.

array dimension

See Also:

The dimension of an array is the number of items in a single element's key. For example, a two-dimensional array has keys that look like this: `myArr[4,2]`

An array can be displayed as a grid with the same dimension. For example, a one-dimensional array can be displayed as a simple list, and a two-dimensional array can be displayed as a set of rows and columns.

array

See Also:

A list of elements that are all stored in a single variable or property.

ASCII

See Also:

American Standard Code for Information Interchange. ASCII represents each text character by a number in the range 0 to 127. This number is the character's ASCII value.

(An ASCII value is also used loosely, to refer to the numeric equivalent of special characters that aren't in the ASCII set.)

associative array

See Also:

An array whose keys can be any string, not just integers. Associative arrays let you name each element rather than numbering it.

attribute

See Also:

Text that modifies and is included inside a tag. An attribute consists of a name and value, joined with an equals sign. Tags can have any number of attributes.

AU

See Also:

AUdio. A format for sound files that is often used on Unix systems. This format is also called ômu-lawö.

The name of a file in this format usually ends with ô.auö.

audio clip

See Also:

An object that holds sound data. Audio clips are stored in a stack, instead of being stored in separate files.

automatic database query

See Also:

A collection of settings for connecting to a SQL database and fetching a selected set of records.

You set up automatic database queries in the Database Query Manager.

AVI

See Also:

Audio-Video Interleave. A format for video files produced by Microsoft's Video for Windows. The name of a file in this format usually ends with `.avi`.

background

See Also:

A set of cards in a stack, all of which contain the same group.

backScript

See Also:

A script that has been inserted into the message path after all other objects, and receives all messages that are not intercepted earlier by other objects in the message path.

behavior

See Also:

The way things on the screen act when the user interacts with them by typing, clicking, dragging, and so on. Part of the look and feel of a user interface, along with appearance.

binary data

See Also:

Data consisting of a string of bits. Non-text data.

binary file

See Also:

A file that contains data other than plain text.

binary

See Also:

The base-two number system, or a number represented in base two. Binary numbers are combinations of two digits: one and zero.

Also, an operator that takes two operands.

BinHex

See Also:

A format for transforming binary files by encoding them as ASCII text for transfer over 7-bit links (which garble non-text files).

The names of BinHex files usually end in `.hqx`.

bit depth

See Also:

The number of bits required to describe the color of a single pixel. The bit depth determines how many different colors can appear on a screen.

For example, a screen whose bit depth is 4 can display 2^4 (16) colors.

bit

See Also:

Binary digit: one digit of a binary number, a one or zero. The smallest unit of information.

The number of bits in a binary number specifies how large the number can be. For example, an 8-bit binary number can equal any of 256 (2^8) values.

bitmap

See Also:

A picture consisting of rows and columns of pixels, with a color assigned to each pixel.

(If the picture is black-and-white, each pixel can be represented by one bit: 1 for black, 0 for white.)

bitwise

See Also:

An operation done on each individual bit in a number's binary representation, rather than on the entire number.

black-and-white

See Also:

A picture consisting only of black and white with no other colors, or a screen that can show only black and white and cannot show any other colors.

BLOB

See Also:

Binary Long Object. A SQL database data structure capable of holding more than 64K of binary data.

block comment

See Also:

A comment that spans multiple lines.

blocking

See Also:

An operation that causes the current handler to stop and wait until the operation is completed.

BMP

See Also:

BitMaP. A format for bitmapped pictures, developed by Microsoft and usually used on Windows systems.

boolean

See Also:

A value that can be either true or false.

breakpoint

See Also:

An instruction to stop execution of a handler at a certain point and enter the debugger.

Browse tool

See Also:

Tool used in the Revolution development environment to interact with a stack, click buttons, type text in fields, and test stack operations.

When the Browse tool is being used, the cursor is in the shape of a pointing hand. You change this shape by setting the defaultCursor property.

browse

See Also:

To interact as a user with a program or web page: to use the program, rather than change or reprogram it.

browser

See Also:

An application used to get, display, and use web pages and other Internet resources.

buffer

See Also:

An area of memory where data is stored for quick retrieval. Data may be pre-processed while in the buffer so it can be used quickly when needed.

build

See Also:

To create a standalone application.

built-in command

See Also:

A command that is part of the Transcript language.

built-in function

See Also:

A function that is part of the Transcript language.

built-in message

See Also:

One of the messages built into Revolution, which are sent when the user performs an action such as clicking or typing.

built-in property

See Also:

A property that is part of the Transcript language. Most built-in properties affect the appearance or behavior of the object.

button menu

See Also:

A menu built from the contents of a button, and displayed when the button is clicked.

Button menus can be displayed as pulldown or popup menus, combo boxes, or dialog tabs.

button

See Also:

An object you click to control some aspect of behavior.

Revolution buttons include push buttons, radio buttons, checkboxes, menu items, and menus.

byte order

See Also:

The order in which an operating system stores bytes when using multi-byte data types.

When transferring data between two systems that use different byte orders, it may be necessary to convert from one byte order to the other.

byte

See Also:

A unit of data equal to 8 bits. A byte can have any of 256 (2 to the 8th power) values. One byte of storage can hold one character of text.

(Additional units include the nybble [4 bits, or half a byte], playte [2 bytes], plattyr [4 bytes], and crayte [8 bytes]. However, these units are much too silly to be referenced in this glossary.)

cache

See Also:

A temporary storage area on disk or in memory.

Also, to place information in a cache for later use.

call

See Also:

To invoke a handler; to cause a handler to execute. In particular, to cause a function or `getProp` handler to execute and return a value.

callback

See Also:

A message that is sent at a specific point in a process. Callbacks are usually used to notify an object about the progress of a process that the object started.

For example, a player might send a callback message when the movie reaches a certain frame. Or the Internet library might send a callback to the object whose script started a file transfer when the transfer is completed.

caller

See Also:

The handler that invoked whichever handler is currently running.

For example, if Handler A uses a custom command which is defined by Handler B, then Handler A is the caller of Handler B.

Carbon

See Also:

One of two available frameworks and sets of programming interface for creating native OS X applications. (The other is called Cocoa.) Revolution creates Carbon applications for OS X, and the Revolution development environment for OS X is a Carbon application.

card control

See Also:

A control (such as a field or button) that is not part of a group and therefore appears on only one card of a stack.

card

See Also:

An object that can contain any combination of buttons, fields, groups, scrollbars, images, graphics, players, and EPS objects.

One card of a stack can appear at a time.

cascading menu

See Also:

A menu that appears when you choose a menu item in the parent menu.

case-insensitive

See Also:

Without regard to whether letters are uppercase (capitalized) or lowercase (not capitalized).

case-sensitive

See Also:

Treating an uppercase (capitalized) letter as a different character than its lowercase equivalent.

cell

See Also:

In a table field, the intersection of a row with a column; one piece of data. You identify a cell with its column number and row number, separated by a comma.

CGI

See Also:

Common Gateway Interface: a method to transfer data between a web server and a program that does something with the data. Also, a program designed to handle data from a web server.

channel

See Also:

One of the sets of values that makes up an image. A channel includes one value for each pixel in the image.

For example, a standard color image has three channels—red, green, and blue—and each pixel has a number for the intensity of red (and one for the intensity of green and of blue—three in all). The set of red intensities for all the pixels makes up the image's red channel.

character set

See Also:

The set of all characters that can be used on a particular operating system, along with the mapping between characters and ASCII values.

character

See Also:

A single text symbol: a letter, number, punctuation mark, or control character.

Characters can be single-byte or double-byte (Unicode). When using the word `character` in a chunk expression, single-byte characters are assumed.

checkbox

See Also:

A special button type consisting of a box with a mark in it. Clicking the button turns the mark on or off.

checksum

See Also:

A value calculated from a block of data. Checksums are used to make sure a block of data has not been changed or corrupted.

For example, after a file transfer, the original checksum can be compared to a new checksum computed on the transferred copy of the file. If they don't match, the file was altered during transmission and needs to be re-sent.

child node

See Also:

A node beneath another node in an XML tree. Any node can have any number of child nodes.

A child node corresponds to an element that is enclosed in the parent node.

chunk expression

See Also:

A specific description of part of a container. A portion of text.

chunk

See Also:

A part of the text in a container.

clipboard

See Also:

The area where cut or copied data is stored.

close box

See Also:

The small box in a window's title bar that, when clicked, closes the window.

codec

See Also:

COmpressor/DECompressor. A codec is a method for storing digital audio or video data. There are many codecs available with QuickTime.

collapse box

See Also:

Box at the right end of a window's title bar (on Mac OS systems) that hides everything but the title bar when you click it. Clicking the collapse box again reveals the window contents.

collapse

See Also:

To hide all of a window except its title bar in order to move it out of the way. Clicking the collapse box in the title bar restores the window to its original form.

color palette

See Also:

The set of all colors used in an image.

color reference

See Also:

Any valid way of describing a particular color that Revolution can understand.

Colors can be referred to by color name, HTML-style specification, or as three comma-separated numbers between zero and 255 (one for each of red, blue, and green).

color table

See Also:

The set of all available colors that can be displayed on the screen.

column

See Also:

In a database, all the values of a particular field from all records in the database.

combo box

See Also:

A control consisting of a drop-down scrolling list, with a box at the top holding the current selection. You can type text into the box or select an option from the list.

Command key

See Also:

A modifier key on Mac OS and OS X systems, located at the right of the spacebar.

The Command key is used for some actions, such as shortcuts for menu items, that require the Control key on Unix and Windows systems.

command line

See Also:

A type of user interface controlled by typing commands and waiting for responses.

Also, the statement you type into a command-line interface to start up a program.

command

See Also:

A directive that causes Revolution to perform an action (such as open a window, put text in a field, or beep).

Revolution has many built-in commands, which are described in the Transcript Dictionary. You can also create your own custom commands.

comment

See Also:

A remark placed in a script to explain the script or part of it. Revolution ignores comments when executing a handler.

Common library

See Also:

One of the Revolution custom libraries, which implements functions and commands that are shared between other custom libraries or used for miscellaneous purposes.

compile error

See Also:

A problem or mistake in a handler that prevents it from being compiled.

Most often, compile errors are caused by incomplete or incorrect control structures (such as a repeat structure with no end repeat) or by punctuation errors (such as failing to quote a literal string).

compile

See Also:

To translate source code—the statements you type into a script—into executable code that can be understood by the computer.

Revolution compiles a script when you click **Apply** in the script editor, close the script, or use the **set** command to change the script.

compress

See Also:

To remove redundancies in data, making the data smaller.

concatenate

See Also:

To attach two pieces of data together.

conditional

See Also:

A programming structure that checks a condition, and then takes appropriate action depending on whether the condition is true or false.

console

See Also:

The window that appears when you run a DOS program on a Windows system.

constant

See Also:

A value that has a name, like a variable, but that doesn't change.

constrain

See Also:

To limit an action.

For example, holding down the Shift key while drawing a rectangle constrains its height to be the same as its width.

container

See Also:

Something that holds data that can be retrieved.

In Revolution, containers include variables, URLs, fields, buttons, and images.

contextual menu

See Also:

A popup menu whose menu items apply to the object or area that the user clicked.

Contextual menus usually appear when you Control-click (on Mac OS and OS X systems) or right-click (on Unix and Windows systems).

contiguous

See Also:

Next to each other; adjacent to each other.

control structure

See Also:

A statement (or set of statements) that determines the order in which other statements are executed.

control

See Also:

An object that can be placed on a card. Controls include buttons, fields, images, graphics, players, EPS objects, scrollbars, and groups.

controller bar

See Also:

The area at the bottom of a QuickTime movie containing controls to play, stop, or move frame-by-frame through the movie.

creator signature

See Also:

On Mac OS and OS X systems, a four-character signature attached to each file. The creator signature identifies the application that should be launched when the file is opened.

cross-platform

See Also:

Able to be used on more than one type of operating system and computer.

current card

See Also:

The card currently being displayed in a stack window.

current folder

See Also:

The folder where saved files will be placed by default. Relative file paths are interpreted as starting from the current folder.

The current folder is specified by the defaultFolder property.

current stack

See Also:

The stack thatÆs being worked on, usually the active window. The current stack is specified by the defaultStack property.

Also, the stack that contains the object whose script is currently executing.

cursor**See Also:**

The arrow, hand, or other small picture that shows you where the mouse pointer is on the screen.

custom command

See Also:

A command whose behavior is defined in a message handler that you write, as opposed to a built-in command.

custom function

See Also:

A function whose behavior is defined by a function handler that you write, as opposed to a built-in function.

custom property set

See Also:

A set of custom properties you define and name.

Each custom property set is independent of the others, and the same property can have different values in different sets.

custom property

See Also:

An object property you define. A custom property can hold any data you want to associate with the object.

Darwin

See Also:

Operating system that is the underlying core of OS X, written by Apple Computer. Darwin is a variant of Unix.

It is possible to run Darwin by itself, without the Aqua user interface and other services that make up OS X. The Darwin version of the Revolution engine supports only command-line scripting.

data fork

See Also:

The part of a Mac OS file that holds unstructured data (as opposed to the resource fork).

database driver

See Also:

Software that acts as a translator between an application and a SQL database.

database field

See Also:

The smallest part of a database. One kind of information is stored in each database field, one value for each record in the database.

Database library

See Also:

The Revolution custom library that supports connections to MySQL, PostgreSQL, ODBC, Valentina, and Oracle (Revolution Enterprise only) databases.

Database Query Builder

See Also:

Part of the Revolution development environment, the window where you create automated database queries. You use these queries to display data from SQL databases in fields and images.

To display the Database Query Builder, choose Tools menu>Database Query Builder.

database

See Also:

A structure that organizes and stores a collection of information as a set of records, with each record containing all the information pertaining to one subject of the database.

datagram

See Also:

A self-contained clump of data that contains information about the route it should take to the system intended to receive it.

debug

See Also:

To find and fix anomalies in a handler that cause errors or unsatisfactory results when the handler is executed.

debugger

See Also:

A feature of Revolution that helps you check scripts for errors.

To use the debugger, you click `Debug` in the error window.

decimal point

See Also:

The character that divides the integer part of a number from the fractional part, usually indicated by a period (.). For example, in the number 22.50, the decimal point is the third character.

decimal

See Also:

The base-10 number system, or a number represented in base 10. Decimal numbers are our normal everyday numbers.

declare

See Also:

To create a new variable or constant, or to make an existing variable available to the current handler.

decode

See Also:

To translate encoded data back to its original format.

decrypt

See Also:

To unscramble encrypted data, recovering the original text. Decrypting data requires that you have the correct password.

default button

See Also:

The button in a dialog box that is automatically clicked if the user presses the Return or Enter key.

The default button is typically indicated by a different color or a thicker outline.


default

See Also:

The setting that is used if you don't specify any other setting.

degree

See Also:

Measurement unit for angles, also shown by the symbol . There are 360 degrees in a circle.

delimit

See Also:

To use a particular character to mark the beginning or end of a piece of data, or separate pieces of data from each other.

For example, a string is delimited by double quotes ("): a double quote is placed at the beginning and end of the string to separate it from the surrounding code.

delimiter

See Also:

The character that marks the start or end of a piece of data, or separates two pieces of data from each other.

dereference

See Also:

To access a value that a variable points to.

For example, if a variable contains a property name, the variable is said to reference the property. Dereferencing the variable's value means getting the value of the referenced property.

desktop

See Also:

The whole area of the screen, underneath any windows; the part of the screen where there are no windows, menus, icons, or other elements.

development environment

See Also:

The user interface Revolution provides for development, including the standard menus, message box, debugger, tool palettes, and so on.

device driver

See Also:

Software that acts as a translator between the computer system and a peripheral device. The driver converts software requests into hardware actions and hardware actions into software results.

dialog box

See Also:

A window that communicates information or requests information.

dimmed

See Also:

Displayed in a special way that indicates that the dimmed item is disabled. Usually, dimmed objects are grayed out: gray is substituted for black when drawing the button, menu item, or other disabled portion of the screen.

disabled

See Also:

Unable to be chosen or used, usually temporarily. For example, the "Next" button might be disabled if the user is already looking at the last entry.

A disabled part of the screen is usually dimmed to the user that the object is unavailable for use.

disclosure triangle

See Also:

A small triangle or arrow in a list or at the edge of a window or palette that, when clicked, flips over and expands the window to show additional options.

Display PostScript

See Also:

An implementation of the PostScript page display language specialized for screen display.

Distribution Builder

See Also:

Part of the Revolution development environment, the window where you create a standalone application or a stack distribution.

To display the Distribution Builder, choose File menu>Build Distribution.

dither

See Also:

To create the illusion of additional colors in an image, by using dot patterns made of the available colors.

For example, if orange is not available on the screen, a pattern of alternating red and yellow dots can be used instead, creating an approximation of orange.

DLL

See Also:

Dynamic Link Library. A set of custom functions that have been written in a programming language and compiled for use on Windows systems.

DNS

See Also:

Domain Name Service. The service that translates domain names into the corresponding Internet addresses.

dock

See Also:

On OS X systems, an area of the screen that contains icons for running applications, minimized windows, and other items.

document

See Also:

A file created by an application.

domain name

See Also:

The name of a computer connected to the Internet. A domain name corresponds to one or more numeric IP addresses.

domain

See Also:

The ownership status of an object. A card object is in the card domain; an object that is part of a shared group is in the background domain.

double quote

See Also:

The double straight quote character, ASCII 34 (").

Double quotes are not interchangeable with ôsmart quotesö or curved quotes in Transcript statements. A double quote is also not the same as two single quote characters (").

double-byte character

See Also:

A character from a character set that supports more than 256 characters; a Unicode character.

The numeric equivalent of Unicode characters is between zero and 65,535, for 65,536 total characters. 65,536 is 2^{16} , so it takes 16 bits (two bytes) to store a Unicode character.

double-byte font

See Also:

A font in which each character is represented by 2 bytes. Languages such as Japanese, Chinese, and Korean, which contain more symbols than can be represented by 256 code points, require double-byte character sets.

Double-byte fonts usually also contain a full complement of alphabetic characters occupying the first 256 positions in the font, so you can display Roman-alphabet text and Unicode text using the same font.

double-click

See Also:

To click twice on a mouse button in rapid succession.

download

See Also:

To receive a file from another system.

drag and drop

See Also:

A method of moving or copying data (such as file icons or text) in which the user selects the data, then drags it to a destination and drops it there by releasing the mouse button.

drawer

See Also:

On OS X systems, a subsidiary window associated with a parent window. A drawer appears at one of the parent window's edges, sliding out from under the parent window, and remains attached to the parent window until it is closed.

DSN

See Also:

Data Source Name. A specification that identifies a particular database, set up in the ODBC manager on your system.

A DSN contains the name of the database, its host, the user name and password, and other information. You can specify a DSN instead of specifying the identifying information directly when connecting to a database via ODBC.

DTD

See Also:

Document Type Definition. A collection of XML declarations that define the structure of a particular XML document.

The DTD defines which tags are permitted, as well as their relationships.

editable window

See Also:

The normal style of window, as opposed to palettes or modal and modeless dialog boxes. Controls in editable windows can be selected with the pointer tool.

element

See Also:

A single member of an array. You refer to an element by the array name and the element's key in the array.

Emacs

See Also:

Popular text editor found mostly on Unix systems.

encode

See Also:

To translate data into a different format.

encrypt

See Also:

To scramble data so that it can't be read by anyone without the proper password.

end-of-line marker

See Also:

The character or characters used to mark the end of a line in a text file.

Different operating systems use different characters to mark the end of a line: Mac OS and OS X use a return character (ASCII 13), Unix systems use a linefeed character (ASCII 10), and Windows systems use a return followed by a linefeed.

engine

See Also:

The back-end software that powers the Revolution development environment, as well as standalone applications created with Revolution.

The engine includes the Transcript compiler, the object hierarchy, and object display routines.

environment variable

See Also:

Information the operating system provides about the system or the user. Environment variables are used only on OS X and Unix systems.

EOF

See Also:

End of file. The last character in a file.

eon

See Also:

The period of time used as the baseline for a computer system or program. Since only finite numbers can be represented in a computer, operating systems and programs that deal with dates must use a starting and ending date; dates outside that range require special handling.

Revolution uses midnight, January 1, 1970 as the start of the eon. Date and time functions are computed from that date.

EPS

See Also:

Encapsulated PostScript. A file that contains PostScript commands and a non-PostScript preview image (used for display on non-PostScript devices).

error message

See Also:

Information about an error that occurred.

An error message may be displayed by a program or handler. Or, if the program or handler that had the error was called by another program or handler, the error message may be returned to the caller instead.

error

See Also:

A problem or mistake in a program or handler, which prevents it from being compiled (compile error) or from running (execution error).

Escape key

See Also:

A key commonly used to stop a program's operation. On most keyboards, the Escape key is near the upper left corner.

evaluate

See Also:

To calculate the final result of an expression; to turn an expression into a single value.

Evaluation Edition

See Also:

A copy of Revolution that has been set up for a thirty-day trial. The Evaluation Edition has the full capabilities of the Revolution Enterprise Edition as long as the trial period lasts.

event

See Also:

An occurrence that causes a program to do something. An event can be a user action (like a mouse click) or a program action (like the completion of a download).

exception handling

See Also:

A programming structure that attempts to execute one or more statements, and then takes appropriate action if there is an execution error.

execute

See Also:

To run a handler or to perform a command.

execution error

See Also:

A problem or mistake in a handler that appears when the handler is executed.

Also, the text Revolution displays when it encounters an execution error.

exit

See Also:

To stop execution of a handler or program.

explicit focus

See Also:

A mode in which you have to click a window to make it the active window.

Also called "active focus" or "click-to-type focus".

export

See Also:

To place data in a file that can be read by other applications.

expression

See Also:

A value, a source of value, or a set of values combined with operators, that can be taken as a single whole.

extension

See Also:

One or more characters following a period at the end of a file name, that indicate what type of file it is.

Windows systems determine which application owns a file based on the file's extension. OS X systems determine a file's type based on its extension.

external

See Also:

A custom command or function that has been written in a programming language and compiled.

An external can be used to implement a command that Transcript does not support directly, thus extending the language.

factor

See Also:

The first fully-resolvable part of an expression: a value, source of value, or combination of values that does not contain any binary operators outside parentheses.

fat application

See Also:

On Mac OS systems, an application that includes both 68000 and PowerPC code (and can therefore be run on either PowerMacs or older Macs without loss of performance).

field

See Also:

An object that contains text. Revolution fields can be editable or locked, and include rectangular fields, scrolling fields, and clickable list fields.

file dialog box

See Also:

A dialog box that displays the files in a folder, and lets you move from folder to folder. You use file dialog boxes to select a file or to name a new file.

file path

See Also:

The name and location of a file, ending with the file name. A file path may be absolute (starting from the disk the file is on) or relative (starting from the current folder and going up, down, or sideways in the folder hierarchy as necessary).

file

See Also:

A collection of data stored on disk under a single name and, in graphical user interfaces, a single icon.

Files include text files, programs, stack files, and many other types.

firewall

See Also:

A system with special security features, which intercepts all traffic between a local network (such as a corporate network) and the rest of the Internet.

Firewalls are used to prevent crackers from gaining access to computers on the local network.

flag

See Also:

A parameter that can take one of two values and indicates whether a setting is on or off.

focus

See Also:

The active status of a window or control, in which the window or control can receive input from the keyboard. The control that currently has the focus receives any characters the user types.

folder

See Also:

A collection of files and other folders.

The synonym *directory* is more common among Unix users. The term *folder* is more often used by Mac OS, OS X, and Windows users.

font

See Also:

A typeface available on the computer. A design for all the characters in the system's character set.

format

See Also:

To specify the appearance or layout of text or numbers.

Also, a standardized pattern for data in a file; a standard type of file.

frame

See Also:

One of the images in the sequence of images that makes up an animation or video.

frontScript

See Also:

A script that has been inserted into the message path before all other objects, and receives all messages before any other objects in the message path.

FTP

See Also:

File Transfer Protocol. The protocol used for transferring files via the Internet to and from a file server.

function call

See Also:

A statement containing a function name (and any needed parameters).

Also, the act of getting the value of a function; the use of a function.

function handler

See Also:

A Transcript handler you write that defines a custom function. Function handlers begin with the function control structure.

function

See Also:

A piece of code that computes a value and returns the value to the handler that called it.

gamma

See Also:

The relationship between intensity and luminance of color.

Gamma varies between monitors and between operating systems, so color images may look different—brighter or more washed out—on different systems.

Geometry library

See Also:

The Revolution custom library that supports the Geometry pane in the property inspector. The `revCacheGeometry` and `revUpdateGeometry` commands are part of the Geometry library.

geometry management

See Also:

The process of moving and resizing controls in a stack window when the window size is changed.

You can perform geometry management in a `resizeStack` handler, or use the Geometry pane in each object's property inspector to specify geometry management settings.

getProp call

See Also:

A request for the value of a custom property of an object.

If the object or one of its owners has a getProp handler for that custom property, the getProp call causes the handler to be executed.

getProp handler

See Also:

A Transcript handler you write that defines optional actions when a custom property's value is requested. GetProp handlers begin with the getProp control structure.

GIF

See Also:

Graphic Interchange Format. A bitmapped color picture format.

global

See Also:

Pertaining to the whole application environment, rather than a single handler or object.

A global variable's value is available to any handler; a global property affects the overall environment, not just one object.

graphic

See Also:

An object that consists of a shape that can be altered, scaled up or down, rotated, or transformed in other ways. Unlike an image, a graphic is a defined shape rather than a bitmap.

greedy

See Also:

A form of regular expression that tries to include as long a string of text as possible while still matching the pattern.

group

See Also:

An object that contains one or more buttons, fields, scrollbars, images, players, EPS objects, or graphics, or other groups.

A group can be placed on any card in a stack, and can optionally have borders or scrollbars.

group-editing mode

See Also:

A state in which only the group being edited is shown, you can select and edit individual controls in the group, and any new controls you create are automatically added to the group.

grouped control

See Also:

A control, such as a button or field, that is part of a group.

grouped text

See Also:

A part of the text in a field whose text style is ôlinkö.

handle

See Also:

To intercept a message or function call. An object handles a message or function call if the object's script contains a handler for it.

handler

See Also:

A group of Transcript statements that begin with an on, function, setProp, or getProp statement and end with an end statement. The basic unit of the Transcript language.

hexadecimal

See Also:

The base-16 number system, or a number represented in base 16. The letters A-F are used as digits representing the numbers 10-15.

Hexadecimal numbers are often used in programming because a single hex digit represents exactly four bits, and two hex digits represent a byte.

highlight

See Also:

To change the appearance of a button or other object when itÆs clicked, in order to make it stand out on the screen.

home directory

See Also:

On OS X and Unix systems, the folder that belongs to a specific user.

A user's home directory has the same name as the user name.

host byte order

See Also:

The order in which the local system (the `host`) stores bytes when using multi-byte data types. This depends on the operating system and microprocessor your computer has.

host

See Also:

A computer system connected to another system.

HTML

See Also:

HyperText Markup Language. The language used for creating web pages.

HTTP

See Also:

HyperText Transfer Protocol. The protocol used for transferring web pages from a web server to a system accessing the web page.

HyperCard

See Also:

A graphical authoring environment from Apple Computer. Its scripting language is called HyperTalk.

hypertext

See Also:

Text containing links that can be clicked in order to display further information.

iconify

See Also:

To change a window into a desktop icon in order to move it out of the way. (On Windows systems, the term "minimize" is usually used instead.)

IDE

See Also:

An acronym for Integrated (or Interactive) Development Environment. An all-in-one environment for developing applications.

idle

See Also:

Not doing anything. A program is idle if it is not currently executing any statements.

image

See Also:

An object that contains a bitmapped picture.

implicit focus

See Also:

A mode in which the window under the mouse pointer is the active window and receives all typed characters, even if that window is not frontmost.

Also called *ôpointer focusö* and *ôsloppy focusö*.

import

See Also:

To bring data into a file from another file.

inheritance

See Also:

The structure under which some objects can use the behavior defined for other objects.

In Revolution, objects inherit some of the properties of their owner. For example, all the objects on a card inherit the card's handlers.

insertion point

See Also:

The location in a field or text box where text you type will appear. Also, the blinking vertical bar that marks that location.

integer

See Also:

A number that is a multiple of 1. Examples of integers include 6, 2200, -145, and 3×10^8 .

Numbers which are fractions or contain a fractional part (such as $1/2$ or 22.87) are not integers.

Internet library

See Also:

The Revolution custom library that supports use of the ftp and http protocols.

inverse

See Also:

The counterpart of a function or operator which returns the original argument. The inverse undoes the function or operator.

For example, square root is the inverse of squaring: if you square any number and then take the square root, the result is the original number.

IP address

See Also:

The numeric address of a computer system, expressed in the form X.X.X.X where each X is a number with between 1 and 3 digits.

ISO 8859

See Also:

The character set used by Unix and Windows systems.

item

See Also:

A chunk of text that is delimited by the character in the itemDelimiter property. (By default, the item delimiter is a comma.)

iteration

See Also:

One repetition of a repeat loop; once through the loop.

Also, a method of computation that works through each of a set of values.

JPEG

See Also:

Joint Photographic Experts Group. A bitmapped color picture format.

key binding

See Also:

The mapping between keys (or key combinations) and the action that pressing the key performs.

key combination

See Also:

A keypress using a modifier key along with another key.

key frame

See Also:

A frame in an animation that is specified by the creator of the animation, rather than generated automatically by tweening.

key

See Also:

A number or word that can be used to select one element of an array. Each element of the array has a key, and you use the key to specify the corresponding element.

keyboard equivalent

See Also:

A key (or key combination) that acts as a shortcut to choose a menu item or button.

keycode

See Also:

The internal numeric identifier for a key on the keyboard. Each physical key on the keyboard can be identified by its keycode.

keyword

See Also:

A word that is part of the Transcript language, but is not a built-in command, function, property, or message name, a control structure, operator, or object type.

landscape

See Also:

The orientation of a printed page that is wider than it is tall.

launch

See Also:

To start up a program.

layer

See Also:

The position of a control relative to the front and back; the order in which controls are stacked on the card.

library

See Also:

A repository of code that is available to be used by other routines. Revolution includes several custom libraries you can include in your applications.

You can make the script of any stack into a library with the `start using` command, or make any object's script into a library using the `insert script` command.

line

See Also:

A chunk of text that is delimited by the character in the `lineDelimiter` property. (By default, the line delimiter is a return character.)

Also, a straight line graphic, or a line drawn with the paint tools.

list field

See Also:

A clickable list, where clicking anywhere in a line selects the entire line.

literal string

See Also:

A group of characters used exactly as they are, rather than as the name of an object, property, function, or variable.

Literal strings are usually enclosed in double quotes (").

load

See Also:

To receive and display a file.

loaded into memory

See Also:

Present in memory, able to be referenced and used.

A stack file (and all its objects) is loaded into memory when any of its stacks is opened. You can also load a stack file into memory by referring to any property of its main stack. Stacks remain loaded into memory until they are deleted from memory or the application quits.

local file

See Also:

A file that's located on the system where the application is running.

A local disk is one that's connected directly to your system, not connected via a local network or the Internet.

local property

See Also:

A property which applies to the general environment instead of a single object, but which is only effective for the current handler.

local variable

See Also:

A variable that is accessible in only one handler and whose existence ends when the handler stops executing.

lock

See Also:

To set something up so it cannot be changed. For example, a locked field's text cannot be edited.

logical

See Also:

A type of operator whose resulting value is either `true` or `false`. Also called a Boolean operator.

look and feel

See Also:

The overall appearance and behavior of a computer system.

loop

See Also:

A control structure that repeats a set of statements.

lossless compression

See Also:

A type of compression technique, used for pictures, that retains all the picture's original information. Unlike lossy compression methods, lossless compression allows the picture to be repeatedly compressed and uncompressed without reducing the quality of the image.

lossy compression

See Also:

A type of compression technique, used for pictures, that discards some of the data in order to compress it better. The compression used to store JPEG files is lossy.

Mac OS

See Also:

An operating system from Apple Computer that runs on Macintosh computer systems. (See also OS X, the successor operating system to Mac OS.)

MacBinary

See Also:

A format for transferring a Mac OS file from one system to another. The MacBinary format preserves the data fork and resource fork of the file, as well as additional information such as the file's type signature and creator signature.

main stack

See Also:

A stack that contains another stack.

Each stack file contains one main stack, and may optionally contain one or more substacks.

mask

See Also:

A filter that allows only certain values. All values that don't conform to the mask are discarded.

In an image, a mask is used to remove transparent pixels.

master profile

See Also:

The default property profile of an object. The master profile holds the default settings for the object's properties, which are used if no other profile is active.

matched

See Also:

Containing both the start and end word or symbol.

For example, an expression with matched parentheses contains a close parenthesis $)$ for each open parenthesis $($.

maximize button

See Also:

The small box in a window's title bar that, when clicked, maximizes the window to fill the screen.

maximize

See Also:

To enlarge a window to the full size of the screen. (On Mac OS systems, the term *zoom* is usually used instead.)

menu bar

See Also:

The set of menus displayed at the top of the screen (on Mac OS and OS X systems) or the top of the window, containing all available menu items.

Menu Builder

See Also:

Part of the Revolution development environment, the window where you create a cross-platform menu bar.

To display the Menu Builder, choose Tools menu>Menu Builder.

menu item

See Also:

One of the available choices (lines) in a menu.

menu

See Also:

A list of actions or options that the user can choose from.

message box

See Also:

A window into which you can enter Transcript commands for immediate execution.

message handler

See Also:

A handler you write that either handles a built-in message (such as `mouseUp`) or defines a custom command. Message handlers begin with the `on` control structure.

message order

See Also:

The order in which messages are sent in response to a user action (such as a mouse click) that triggers more than one message.

message path

See Also:

The order in which objects are given the opportunity to respond to a message.

message watcher

See Also:

Part of the Revolution development environment, a window that displays the names of messages as they are sent.

To display the message watcher, choose Development menu>Message Watcher. In the debugger, choose Debug menu>Message Watcher.

message

See Also:

An instruction sent to an object. If the object's script contains a message handler with the same name as the message, that message handler is executed.

Meta key

See Also:

Modifier key on Unix systems, usually located to the right of the Control key.

(The Mac OS and OS X equivalent is the Option key. The Windows equivalent is the Alt key.)

MetaCard

See Also:

A graphical authoring environment from MetaCard Corporation. Its scripting language is called MetaTalk.

metal window

See Also:

On OS X systems, a window whose background has a brushed-metal appearance.

millisecond

See Also:

One thousandth of a second.

MIME

See Also:

Multipurpose Internet Mail Extensions. A standard for including data other than plain text in an email message.

minimize button

See Also:

The small box in a window's title bar that, when clicked, minimizes the window to an icon.

minimize

See Also:

To shrink a window to an icon. (On Unix systems, the term *iconify* is usually used instead.)

modal dialog box

See Also:

A dialog box that requires a response before you can use any other window.

modeless dialog box

See Also:

A dialog box that lets you perform another task without responding to the dialog box first.

modifier key

See Also:

A key that modifies the effect of pressing another key, instead of typing a character.

Modifier keys include Shift, Control, Caps Lock, and the Command key (on Mac OS and OS X systems), and the Option key, Meta key, or Alt key.

Motif

See Also:

A graphical user interface for Unix systems.

mouse button

See Also:

Clickable button on a mouse.

Usually, the mouse on a Mac OS or OS X system has one button, the mouse on a Windows system has two, and the mouse on a Unix system has three.

mouse pointer

See Also:

The point on the screen whose position is controlled by moving the mouse.

movie

See Also:

A file that can be displayed by QuickTime. Movies can contain video tracks, sound tracks, or both. QuickTime can also display interactive movies (QuickTime VR) and picture files in a number of formats.

Any movie can be displayed in a player object.

MPEG

See Also:

Moving Pictures Experts Group. A video file format.

navigation message

See Also:

One of the built-in messages Revolution sends automatically when going from card to card or stack to stack. Navigation messages include `openCard`, `closeBackground`, `preOpenCard`, and similar messages.

negative

See Also:

Less than zero. Negative numbers are written with a leading minus sign.

nest

See Also:

To enclose a part within another part of the same kind.

For example, you can nest parentheses in an expression:

$$7 - (3 + (14/3))$$

This expressionÆs parentheses are nested two deep.

network byte order

See Also:

The byte order used by a network. To avoid confusion, all systems on a network use the same byte order when communicating with each other, even if they use a different byte order internally.

For the Internet, by convention, the network byte order is *big-endian*, with the most significant byte first.

NNTP

See Also:

Network News Transfer Protocol. The protocol used to transmit posts on Usenet newsgroups.

node

See Also:

The basic part of which an XML tree is made.

A node in an XML tree corresponds to an element in an XML document.

non-blocking

See Also:

An operation that allows the current handler to continue while the operation is going on.

non-greedy

See Also:

A form of regular expression that matches only the minimum length of text needed.

non-negative

See Also:

Greater than or equal to zero.

non-printable character

See Also:

A character such as backspace that has an ASCII code, but has no visual representation.

null

See Also:

The character whose ASCII value is zero.

numeric

See Also:

A type of operator whose resulting value is a number.

Also, a value consisting of numbers.

object hierarchy

See Also:

The chain of ownership in which one object is owned by another. A card control is owned by the card itÆs on; a grouped control are owned by its group; a card is owned by the stack itÆs in; and substacks are owned by their main stack.

Also, the chain of all the objects above a specific object in the hierarchy.

object reference

See Also:

Any valid way of describing an object that Revolution can understand. Objects can be referred to by name, number, or ID.

object type

See Also:

A category of object; a set of objects that all have the same general properties.

In Revolution, the object types are stacks, cards, groups, buttons, fields, images, graphics, EPS objects, scrollbars, players, audio clips, and video clips.

object

See Also:

A Revolution module capable of receiving messages. One of the building blocks of a Revolution application.

octal

See Also:

The base-8 number system, or a number represented in base 8. Octal numbers are combinations of the digits 0 through 7.

ODBC manager

See Also:

A utility program that manages database connections made via ODBC.

Typically, you use an ODBC manager program to create one or more DSNs with all the information needed to communicate with a database, such as its location and type. You use the DSN with the `revOpenDatabase` function to communicate with that database.

offset

See Also:

A distance from some base point.

Open Database Connectivity (ODBC)

See Also:

A system that allows a program to access any type of database in a standard way.

Open Scripting Architecture (OSA)

See Also:

System for creating and integrating system-wide scripting languages, developed by Apple Computer.

operand

See Also:

An expression on which an operator performs an action; one of the arguments of an operator.

operation

See Also:

An action performed on one or more values.

operator

See Also:

A symbol or word that changes a single value or combines two values to produce a result.

Option key

See Also:

Modifier key on Mac OS and OS X systems, usually located at the left of the Command key.

(The Unix equivalent is the Meta key. The Windows equivalent is the Alt key.)

Oracle Media Objects

See Also:

A graphical authoring environment from Oracle Corporation. Its scripting language is called MediaTalk.

ordinal

See Also:

A number that describes a position in a list (for example, *third*).

OS X

See Also:

An operating system from Apple Computer that runs on recent Macintosh computer systems, the successor to Mac OS.

owner

See Also:

The object that contains another object; the next object in the object hierarchy. A card control is owned by the card itÆs on; a grouped control is owned by its group; a card is owned by the stack itÆs in; and substacks are owned by their main stack.

Also, the application that will open a file when the file is double-clicked.

Also, on OS X and Unix systems, the user who owns a file.

pad

See Also:

To add data to a value in order to come up to an even amount.

For example, if a variable is supposed to hold 10 characters, but the actual data is only six characters, you can pad the variable by adding four spaces to the end to make it the right size.

paint tool

See Also:

One of the tools in the Paint Palette, used to draw bitmapped pictures in image objects.

palette

See Also:

A small floating window usually used to display tools or information about a main window. Also referred to as a windoid or utility window.

pane

See Also:

An area in a window whose content changes depending on the setting you choose. Typically, you switch between panes by choosing the pane you want from a popup menu at the top of the window.

panel

See Also:

A section of a window. A panel in a window may be scrolling or not, and is usually separated by a line or divider from other panels in the same window.

parameter variable

See Also:

A special variable whose name is one of the parameters of the current handler, and whose value is whatever value was passed to the handler for that parameter.

parameter

See Also:

A value passed to a function handler or message handler. Also, the name of a value that can be passed to such a handler.

parent folder

See Also:

The folder that contains another folder.

If a folder called `âAö` is located inside a folder called `âBö`, `âBö` is the parent folder of `âAö`.

parent node

See Also:

The node above a node in an XML tree. A node's parent node corresponds to the element that encloses the node.

pass by reference

See Also:

A way of using parameters where you pass a local variable to the called handler, and changing the parameter within the called handler also changes the variable in the calling handler.

You specify that a parameter is to be passed by reference by preceding its name with the @ character in the called handler's first line.

pass by value

See Also:

A way of using parameters where the values of the arguments are passed to the called handler, but changing the parameters within the called handler has no effect on the calling handler.

If you pass a variable by value to another handler, the other handler can't affect the variable's value in the original handler.

pass

See Also:

To relinquish a message to the next object in the message path.

Also, to send data from one handler to another; to share data between handlers.

password

See Also:

A word or phrase used to lock information against unauthorized access, or to unlock it again.

PBM

See Also:

Portable Bit Map. A black-and-white picture format created for the `pbmplus` image conversion program and usually used on Unix systems.

peripheral device

See Also:

Part of a computer system that lets the computer interact with the user, communicate with another system, or transfer information. Examples of devices include keyboards and mice, modems, printers, and disk drives.

A device is controlled by a piece of software called a device driver.

PGM

See Also:

Portable Gray Map. A grayscale picture format created for the `pbmplus` image conversion program and usually used on Unix systems.

PICT

See Also:

PICTure. A picture format developed by Apple Computer and usually used on Mac OS or OS X systems.

pixel

See Also:

Picture ELe ment. One dot on the screen or in a printed image. Screens usually display between 60 and 120 pixels per inch.

pixmap

See Also:

A picture consisting of rows and columns of pixels, with a color assigned to each pixel.

Also, a data structure in memory that holds such a picture.

platform

See Also:

A combination of computer hardware and software that defines a standard.

Platinum

See Also:

The standard appearance for controls, windows, and menus on Mac OS systems. Platinum is the default, Apple-supplied user-interface theme for Mac OS.

player

See Also:

An object that presents a file containing a QuickTime movie, or audio or video data in one of the formats supported by Quicktime.

plist

See Also:

A specially-formatted XML document that OS X uses to store data about applications.

plugin

See Also:

A custom stack which you install in the Plugins folder (inside the Revolution folder), and which appears in the "Plugins" submenu of the Development menu.

PNG

See Also:

Portable Network Graphics. A bitmapped color picture format.

point

See Also:

A way of specifying a location, consisting of the horizontal distance from the left edge of the screen or window, a comma, and the vertical distance from the top edge of the screen or window.

Also, the act of moving the mouse pointer over a section of the screen in order to act on it.

Also, a measurement of distance used in printing. A point is $1/72$ of an inch.

Pointer tool

See Also:

Tool used in the Revolution development environment to select, move, and resize controls.

When the Pointer tool is being used, the cursor is in the shape of an arrow.

poll

See Also:

To repeatedly check a status (such as whether the mouse button is down or whether a process has completed).

(Polling is usually considered a less desirable technique than sending a message when the status changes, because it takes more processing time to repeatedly check the status and can slow down other programs on the system.)

polygon

See Also:

A shape consisting of vertexes connected by straight lines.

In a regular polygon, all the lines are the same length.

POP

See Also:

Post Office Protocol. A protocol used to retrieve email from a mail server.

popup menu

See Also:

A list of choices that appears where the mouse is clicked.

port

See Also:

A connection into a computer. In a socket identifier, the port is the part of the address that specifies the particular channel on the host. If the IP address is like a telephone number, the port number is like an extension.

Also, to modify an application for a different platform from the one it was originally designed for; or to translate an application to a different programming language than the one the software was originally written in.

Also, an application that has been ported.

portrait

See Also:

The orientation of a printed page that is taller than it is wide.

PostScript

See Also:

A page description language used to lay out information for printing or screen display.

A PostScript file is a file containing information in the PostScript language.

PPM

See Also:

Portable Pix Map. A color picture format created for the `pbmplus` image conversion program and usually used on Unix systems.

precedence

See Also:

The set of rules that determine what order operators are evaluated in, in an expression that has more than one operator.

For example, the $*$ operator has higher precedence than the $+$ operator, so

$$2 + 3 * 4$$

is evaluated as $2 + (3 * 4)$, not $(2 + 3) * 4$.

precision

See Also:

The number of decimal places to which a number can be computed.

prepend

See Also:

To add data to the beginning of a container or file, without removing its current contents.

For example, if a file contains `ôAppleö`, and you append `ôRedö`, the file now reads `ôRedAppleö`.

pretty printing

See Also:

Formatting a script by indenting lines to clarify the control structures in the code.

Pretty printing does not change the way the code works, but it makes it easier to read.

primary key

See Also:

In a database, a field (or combination of fields) that is different for each record in the database. The primary key is used when changing or deleting a record, to uniquely identify the record.

print job

See Also:

A single print task; a single item to be printed. One print job can include multiple pages and multiple cards, and is treated by the printer software as a single printout.

printable character

See Also:

A character that can be displayed as-is on the screen: a letter, digit, or punctuation mark.

Printing library

See Also:

The Revolution custom library that supports text printing. The `revPrintField`, `revPrintText`, and `revShowPrintDialog` commands are part of the Printing library.

process

See Also:

Any program that is currently running.

A process can be an application that interacts with the user, or a system service that operates behind the scenes and is normally invisible to the user.

Profile library

See Also:

The Revolution custom library that supports property profiles. The profile property is part of the Profile library.

prompt

See Also:

Text that requests information from the user.

property inspector

See Also:

Part of the Revolution development environment, a palette window that displays the properties of an object.

To display an object's property inspector, select the object and choose Object menu>Object Inspector.

property profile

See Also:

A set of property settings for an object. You create and change an object's profiles using the Property Profiles pane in the object's property inspector.

property

See Also:

An attribute of an object, or a global attribute of the Revolution environment.

An object's properties consist of its built-in properties, plus any custom properties you have specified for that object.

protocol

See Also:

A standard format and set of rules for transferring a particular kind of data between systems.

proxy server

See Also:

A server that intercepts requests for a particular service (such as a web document).

QuickTime VR

See Also:

Enhanced video system that allows you to display and rotate objects in three dimensions.

QuickTime

See Also:

System for video, sound, and picture display developed by Apple Computer.

Also, a file format that can hold video or sound information (a QuickTime movie).

quoted

See Also:

Surrounded with double quotes (").

radian

See Also:

Measurement unit for angles. 2π radians is one circle, or 360°. One radian is approximately 57°.

radio button

See Also:

A special button type consisting of a circle with a dot in it.

Normally, radio buttons are part of a set, and only one button in the set can be turned on at a time. Clicking a radio button turns that button on, and turns off the rest of the radio buttons in the set.

random

See Also:

Selected by chance from a set.

Also, a set that is in no particular order; scrambled.

range

See Also:

The allowable extent of a value, from the least possible value to the greatest.

read-only

See Also:

Information that can be looked at, but not changed or deleted.

real number

See Also:

A positive or negative number in the real number system; any number. Examples of real numbers include 22.3, -12, 2839.99999, and pi.

A computer representation of a real number is called a floating-point number.

REALbasic

See Also:

An authoring environment from REAL Software, Inc. Its language is also called REALbasic.

recent cards

See Also:

Cards the user has visited between starting up the application and going to the current card.

record set (database cursor)

See Also:

In a database, the set of records returned by a query.

You can use commands in the Database library to move among the subset of records that makes up a record set.

record

See Also:

A set of related values in a database; a complete segment of information, consisting of one or more database fields.

For example, in a database of people, each person's data is stored in a single record.

A record is also called a row. If a set of records is displayed in the form of a grid, each row is a single record and each column is a database field.

rectangle

See Also:

A way of specifying a size and location on the screen, consisting of four integers separated by commas. The four integers are the position of the left, top, right, and bottom edges, in that order.

recursion

See Also:

The process by which a handler calls itself.

(Recursion is a useful programming technique, but may also occur accidentally, resulting in a script error.)

redraw

See Also:

To update a window, or the screen, to display new information.

referenced control

See Also:

A control (image or player) whose contents are located in a separate file, instead of being stored in the stack.

The control's filename property determines where the contents are located.

registry

See Also:

A database of configuraton information, used on Windows systems to associate files with the applications they belong to and to hold various system settings.

regular expression

See Also:

A way of describing a pattern that matches text you are looking for.

regular polygon

See Also:

A shape consisting of vertexes connected by straight lines that are all the same length.

relative coordinates

See Also:

Measurement of a position by its distance from the top and left edges of a window.

relative file path

See Also:

The name and location of a file, starting from the defaultFolder and going up, down, or sideways in the folder hierarchy as necessary to get to the fileÆs location.

remainder

See Also:

What's left after dividing one number evenly into another.

For example, if you divide 5 by 2, the remainder is 1.

Report Builder

See Also:

Part of the Revolution development environment, the window where you create settings for a printed report.

To display the Report Builder, choose Tools menu>Report Builder.

report layout card

See Also:

A card containing one or more report viewer objects.

You use report viewers, along with other objects such as graphics, to lay out a report, then use the Report Builder to set up and print the report.

report viewer

See Also:

A special object used to set up and display data for printed reports.

To create a report, you place one or more report viewers on a card and specify which fields they should display, then use the Report Builder to select the cards to print.

ResEdit

See Also:

Resource editor for Mac OS systems, published by Apple Computer.

reserved word

See Also:

A word that has a special meaning to Transcript (for example, built-in function and command names, keywords, and so forth), and should not be used as the name of a variable, custom property, custom command, or custom function.

reset

See Also:

To restore a value/Es former setting.

Also, to restart a computer.

resource fork

See Also:

The part of a Mac OS file that holds structured data in the form of resources (as opposed to the data fork).

resource

See Also:

A type of data stored in the resource fork of Mac OS files. Each resource is identified by a four-letter resource type and a resource ID number.

resume

See Also:

To start a suspended action up again; take up where the action left off.

return value

See Also:

The result of a function; the value that a function calculates and returns to the handler that called it.

return

See Also:

To send a value back to the handler that called a function.

Also, the carriage return character, ASCII 13.

Revolution custom library

See Also:

One of several libraries of custom functions, commands, or properties that come with the Revolution development environment.

Tip: If your stacks use commands from a Revolution custom library, be sure to check the appropriate option in the Distribution Builder to include the library in your standalone.

root node

See Also:

The topmost node in an XML tree. Each XML tree has one and only one root node.

RTF

See Also:

Rich Text Format. A format for text files, codified by Microsoft Corporation, that includes information about text styles in plain text format.

run time

See Also:

The time of program execution.

When your application does something *at run time*, that something is done when the application is being used by the user, as opposed to ahead of time before you build the application.

scientific notation

See Also:

A way of formatting numbers where the number is represented as a value between one and ten, multiplied by the appropriate power of ten.

scope

See Also:

The extent of places where a variable can be used.

A variable's scope can be either local, script local, or global. A local variable can be used only in the current handler. A script local variable can be used in any handler in a script, but can't be used in handlers that are in other scripts. A global variable can be used in any handler where it's declared.

script debug mode

See Also:

Mode of the Revolution development environment in which the Script Debug window is open. When in script debug mode, executing the breakpoint command halts the current handler.

script editor

See Also:

The window in which you edit scripts. You can open the script editor by selecting the object and choosing Development menu>Object Script, by using the contextual-menu shortcut, or by using the edit command.

script local variable

See Also:

A variable that can be used in any handler in an object's script, but can't be used by handlers in other scripts. You create a script local variable by using the `local` command outside any handler in the script.

script

See Also:

All the Transcript code associated with a particular object.

scrollbar thumb

See Also:

The small box in a scrollbar whose position indicates the amount of scroll.

You can drag the thumb along the scrollbar to change the amount of scroll.

scrollbar

See Also:

An object that displays a quantity, horizontally or vertically.

(Fields and groups can also have scrollbars, but these are part of the field or group, rather than separate objects.)

select

See Also:

To choose text, part of an image, or an object so you can manipulate it or get information about it.

Selected text is normally highlighted. You select part of an image with the Select tool, and the selected portion is shown with a dotted outline. Selected objects are shown with selection handles at each side and corner.

selection

See Also:

The currently selected text or objects. Selected objects and text are normally highlighted in some way.

server

See Also:

A system that provides services to other systems on the same network. For example, a web server provides access to web pages.

setProp handler

See Also:

A Transcript handler you write that defines optional actions when a custom property's value is set. SetProp handlers begin with the setProp control structure.

setProp trigger

See Also:

A request to set a custom property of an object.

If the object or one of its owners has a setProp handler for that custom property, the setProp trigger causes the handler to be executed.

shared control

See Also:

A control that is part of a shared group, and therefore appears on more than one card in a stack. Changes to a shared control made on one card are reflected on each card that shares the control.

shared group

See Also:

A group that appears on more than one card in a stack. Changes to a shared group made on one card are reflected on each card that shares the group.

sheet

See Also:

On OS X systems, a dialog box associated with a specific window. A sheet appears within the window, sliding out from under the title bar. The user can use other windows, but cannot use the sheet's window until the sheet is dismissed.

shell

See Also:

Program that accepts operating-system commands. The program a user uses to interact with the operating system, move files around, launch applications, and so on.

shortcut

See Also:

On Windows systems, a special file type that refers to another file. Double-clicking the shortcut opens the file it refers to.

(The Mac OS and OS X equivalent is the alias. The Unix equivalent is the symbolic link.)

sibling node

See Also:

A node in an XML tree that has the same parent node.

sign

See Also:

The symbol + or - that indicates whether a number is positive or negative.

single-byte character

See Also:

A character from a character set that supports 256 characters; an extended-ASCII character, as opposed to a Unicode character.

The numeric equivalent of single-byte characters is between zero and 255, for 256 total characters. 256 is 2^8 , so it takes 8 bits (one byte) to store an extended-ASCII character.

SMTP

See Also:

Simple Mail Transfer Protocol. Used to send email messages from one Internet-connected computer to another.

SOAP

See Also:

Simple Object Access Protocol. An XML-based set of rules for applications to access and use web services.

socket

See Also:

One of the endpoints of a connection between two processes (which may be running on different systems). Two cooperating sockets, one on each end, form a connection.

sort key

See Also:

The part of a chunk or card thatÆs used to determine the chunkÆs or cardÆs position when sorting.

sort

See Also:

To shuffle units into a new order.

special folder

See Also:

A folder that is a standard part of the operating system, such as the Preferences folder. Special folders may be located in different places and have varying names, depending on the operating system version.

Speech library

See Also:

The Revolution custom library that supports text to speech.

SQL query

See Also:

A request for information from a database, written in SQL (Structured Query Language).

You use a SQL query to specify one or more conditions, and the result of the query is the set of records in the database that match those conditions.

SQL

See Also:

Structured Query Language. A standard language for specifying, retrieving, and updating data in a database. You use the Database library to communicate with databases using SQL.

Note: This documentation pronounces SQL as ôsequelö. Your pronunciation may vary.

stack file

See Also:

A Revolution file, containing a main stack and (optionally) one or more substacks.

stack menu

See Also:

A menu built from buttons placed in a stack.

Stack menus can be displayed in the menu bar, or as popup or pulldown menus or combo boxes.

stack window

See Also:

The window that displays a stack, in which the stack's current card is shown.

A stack can be displayed as an editable window, a palette, or a modal or modeless dialog box.

stack

See Also:

An object that contains cards, which may contain other objects. Each object in Revolution is part of a specific stack.

standalone application

See Also:

An application built with Revolution that can run on its own, without needing the Revolution development environment.

standard error

See Also:

The place where a program sends any error messages it generates.

Usually, the standard error is the command-line window by default. On Unix systems, the standard error can be redirected to a file, a printer, or another destination.

standard input

See Also:

The source where a program looks for incoming data.

Usually, the standard input is the keyboard by default. On Unix systems, the standard input can be redirected to a file, a network connection, or another source of data.

standard output

See Also:

The place where the output, or result, of a program is directed.

Usually, the standard output is the command-line window by default. On Unix systems, the standard output can be redirected to a file, a printer, or another port or device. On OS X systems, the standard output is the Console window.

statement

See Also:

A single programming instruction. A single line of Transcript code that starts with a command.

streaming

See Also:

Playing a sound or movie file as it is being downloaded from a server, rather than waiting for the whole file to be downloaded first.

string

See Also:

A sequence of characters.

Also, a type of operator whose resulting value is a string.

subfolder

See Also:

A folder inside another folder.

If Folder A contains Folder B (and possibly other folders and files), B is a subfolder of A.

subroutine

See Also:

A set of program statements, with a name, that performs a particular task. The subroutine can be called from other routines.

In Transcript, routines are called handlers.

subsidiary window

See Also:

A window that is part of another window or associated with it in some way.

Subsidiary window types include drawers (which slide out from one edge of a parent window) and sheets (which are dialog boxes associated with a parent window).

substack

See Also:

A stack contained in a main stack. Each Revolution file contains one main stack, and may optionally contain one or more substacks.

SuperCard

See Also:

A graphical authoring environment from Solutions Inc. (formerly published by Incwell Ltd.). Its scripting language is called SuperTalk.

suspend

See Also:

To put aside; to put an action on hold.

symbolic link

See Also:

On Unix systems, an alternative file path for a file or directory. Specifying the symbolic link in a file operation uses the file or directory the link refers to.

(The Mac OS and OS X equivalent is the alias. The Windows equivalent is the shortcut.)

synchronize

See Also:

To cause to happen at the same time; to coordinate two or more actions so they start and end at the same time.

syntax

See Also:

The structure of a programming language or of a statement in a programming language.

system window

See Also:

A palette that floats above all other windows (in all running applications) on the system.

Unlike an ordinary palette, a system window is always available regardless of which application is active.

tab stop

See Also:

Distance from the left edge of a field to the position of the insertion point when you press Tab.

tabbed button

See Also:

A button type that lets you select a section of a dialog box or window. A tabbed button looks like a stack of file folders, with several tabs across the top. Clicking a tab displays its section in the dialog box.

table field

See Also:

A spreadsheet-style grid of entries arranged in rows and columns.

table

See Also:

In a database, a set of records that each have the same fields and contain the same kind of data.

You can display a table in the form of a grid. Each row corresponds to a single record. Each column corresponds to one of the fields thatÆs in each record.

tag

See Also:

The structure that marks the start and end of an XML or HTML element. Tags are enclosed in angle brackets and include the tag's name and any attributes it has:

`<tag attribute=value>`

task bar

See Also:

The area of the screen on Windows systems that holds the Start button, icons for open windows and running applications, and other features.

TCP

See Also:

Transmission Control Protocol. Used to establish a connection between two systems that are both connected to the Internet.

template

See Also:

An object that serves as a model for newly created objects of the same type.

temporary memory

See Also:

Memory that is temporarily allocated by Mac OS to applications on request.

On Mac OS systems, each application has a fixed amount of memory, which is set in the application's Get Info window. Temporary memory is in addition to this allocation, and is responsible for applications taking more memory than their allocation.

text file

See Also:

A file that contains only plain, unformatted text.

text to speech

See Also:

A feature of some operating systems that allows the computer to speak a phrase in a human-sounding synthesized voice. Text to speech is supported in Revolution with the Speech library.

tick

See Also:

One sixtieth of a second.

timeout

See Also:

A period of time to wait for an event (such as a connection or the launch of a process) before giving up.

timestamp

See Also:

A date and time attached to an object or file.

title bar

See Also:

The top border area of a window, which contains the window's name.

token

See Also:

One of the words and symbols that make up a Transcript program.

tool tip

See Also:

Explanatory text box that pops up when the mouse pointer hovers over an item on the screen.

tool

See Also:

A mode that lets you perform specific actions (such as selecting objects or drawing lines) when you click in a stack window.

Also, the icon (in a palette of tools) you click to select a tool.

Transcript

See Also:

The language in which Revolution scripts are written.

transfer mode

See Also:

A mathematical operation that determines how an object is drawn in its window.

Some transfer modes make the object partly or completely transparent by combining each pixel's color with the color underneath it.

trap

See Also:

To stop a message from proceeding further along the message path.

trigger

See Also:

To cause an action to happen. In particular, to cause an on or setProp handler to execute.

Also, the set instruction sent to an object that triggers a setProp handler.

tweening

See Also:

The process of automatically generating intermediate frames in an animation.

Tweening lets you specify only the frames where a change takes place, rather than having to generate all the details of each frame.

type signature

See Also:

On Mac OS and OS X systems, a four-character signature attached to each file. The type signature identifies the format of the file.

UDP

See Also:

User Datagram Protocol. A method of sending a self-contained clump of data (a datagram) to another system.

unary

See Also:

An operator that takes one operand.

uncompress

See Also:

To return compressed data to its original state.

undo

See Also:

To reverse the last action taken; to back up a step.

Unicode

See Also:

A standard for representing characters as 2-byte numbers, also called UTF-16. Unicode contains enough characters and symbols to write most human languages.

Unix

See Also:

An operating system originally created at AT&T. Unix runs on a variety of hardware and exists in many versions and varieties.

(For purposes of the Revolution documentation, Linux is considered a variety of Unix. Your geekage may vary.)

unlock

See Also:

To set something up so that it can be changed by the user. For example, an unlocked field's text can be changed.

upload

See Also:

To put a file on another system.

URL scheme

See Also:

A type of URL, specified by the part of the URL before the `://`. URL schemes supported by Revolution include `http`, `ftp`, `file`, `resfile`, and `binfile`.

URL

See Also:

Uniform (or Universal) Resource Locator. The address of a resource or file on the Internet. A URL contains the protocol and location of the resource.

user interface error

See Also:

An internal error caused by a bug in Revolution, not a bug in your scripts.

user interface

See Also:

The way the user controls a computer or computer program, and the way the computer or program reports results to the user.

validate

See Also:

To check whether data is of the correct type, or in the correct format.

value

See Also:

Data that can be used in an operation.

variable watcher

See Also:

Part of the Revolution development environment, a window that displays the current values of variables during script execution.

To display the variable watcher, in the debugger, choose Debug menu>Variable Watcher.

To watch global variables, choose Development menu>Variable Watcher.

variable

See Also:

A container in memory. You can put data of any kind into a variable, and read its contents.

Unlike other containers, variables are not permanent. A local variableÆs existence ends when the handler stops; a global variableÆs existence ends when you quit the application.

vertex

See Also:

The point where two lines are joined at an angle. A corner of a polygon shape.

VFW

See Also:

Video for Windows. System for video and audio developed by Microsoft and used on Windows systems.

video capture

See Also:

The process of getting information from a video camera or other video source, and displaying it on the screen or saving it as a movie file for later playback.

Revolution supports video capture using the Video library.

video clip

See Also:

An object that contains movie data. Video clips are imported into a stack, instead of being stored in separate files like a player.

video grabber

See Also:

The window that video from an external source (such as a video camera) is shown in.

Video library

See Also:

The Revolution custom library that supports video capture from a camera or other video input device.

virtual property

See Also:

A custom property for which an object has a `getProp` or `setProp` handler, but which is never attached to the object.

WAV

See Also:

WAVEform. A sound file format that is often used on Windows systems. The name of a file in this format usually ends with `.wav`.

WDEF

See Also:

Window Definition. A Mac OS and OS X resource that describes how a window is drawn. You can specify custom window styles by using your own WDEF resources along with the decorations property.

web server

See Also:

A system that sends web pages to other systems in response to their requests.

whitespace

See Also:

Characters that don't contain any visible text: returns, linefeeds, spaces, and tabs, or any combination of these characters.

wildcard

See Also:

A symbol that stands for one or more characters.

The process of expanding a wildcard to match all its possible strings is called globbing (after ôglobalö).

Windows

See Also:

An operating system from Microsoft Corporation that typically runs on Intel-based computer systems.

word

See Also:

A chunk of text that is delimited by spaces, tabs, or returns.

XBM

See Also:

X Bit Map. A black-and-white icon format used on some Unix systems.

XCMD

See Also:

External command for Mac OS and OS X systems. A custom command that has been written in a programming language and compiled into a resource of type XCMD.

XFCN

See Also:

External function for Mac OS and OS X systems. A custom function that has been written in a programming language and compiled into a resource of type XFCN.

XML document

See Also:

A text file that contains XML markup.

XML library

See Also:

The Revolution custom library that supports parsing and manipulating XML data and creating XML files.

XML tree

See Also:

An XML document held in memory. An XML tree is a data structure containing all the content in an XML document, arranged to make it easy to traverse parent, sibling, and child relationships between nodes.

XML

See Also:

eXtensible Markup Language. A language used for defining markup languages (such as HTML).

Using XML, you can create custom sets of tags, along with rules for using them, suitable for any sort of data storage and retrieval.

XPM

See Also:

X Pix Map. A color icon format used on some Unix systems.

xTalk

See Also:

One of the family of related scripting languages that includes Transcript, HyperTalk, SuperTalk, Lingo, and others.

XWD

See Also:

X Windows Dump. A bitmapped color picture format used for screenshots on some Unix systems.

zoom box

See Also:

The small box in a window's title bar that, when clicked, toggles the window between normal size and full-screen size.

zoom**See Also:**

To enlarge an object for viewing.

Also, to enlarge a window to the full size of the screen. (On Windows systems, the term "maximize" is usually used instead.)

To align control edges

See Also:

How to block or change the action of arrow keys, How to change the size and position of an object, Shortcut to select multiple controls, Shortcut to nudge the selected control, Shortcut to equalize widths of selected controls, Shortcut to equalize heights of selected controls, Object menu > Align Selected Controls > Left, Object menu > Align Selected Controls > Right, Object menu > Align Selected Controls > Top, Object menu > Align Selected Controls > Bottom, bottom property, left property, right property, top property

Command-Option-arrow key (Mac OS or OS X)

Control-Alt-arrow key (Unix or Windows)

To change a controlÆs properties

See Also:

Shortcut to display a controlÆs property inspector, Shortcut to edit a controlÆs script, Object menu > Object Inspector

Return key

To change message box mode

See Also:

Shortcut to clear the message box, Shortcut to review message box history, Edit menu > Clear, Tools menu > Message Box

Command-, (Mac OS or OS X)

Control-, (Unix or Windows)

To clear the message box

See Also:

Shortcut to review message box history, Shortcut to change message box mode, Edit menu > Clear, Tools menu > Message Box, message box keyword

Command-U (Mac OS or OS X)

Control-U (Unix or Windows)

To close the Error window

See Also:

Shortcut to close the script editor, Shortcut to hide Revolution palettes, Development menu > Suppress Errors, `executionError` property, `scriptError` property

Enter key

To close the script editor

See Also:

(Platform-dependent

Enter key

Control-Return

To apply changes and close the script editor

See Also:

(Platform-dependent

Option-close (Mac OS or OS X)

Alt-close (Unix or Windows)

To apply changes and close the script editor

See Also:

To complete words in scripts

See Also:

(Platform-dependent

Command-number (Mac OS or OS X)

Control-number (Unix or Windows)

To display a contextual menu (Pointer tool)

See Also:

Shortcut to change a control's properties, Shortcut to display a contextual menu, Shortcut to edit a control's script, Shortcut to edit the card script, Shortcut to edit the stack script, Object menu > Object Inspector

Control-click (Mac OS or OS X)

Right-click (Unix or Windows)

To display a contextual menu

See Also:

Shortcut to change a control's properties, Shortcut to display a contextual menu (pointer tool), Shortcut to edit a control's script, Shortcut to edit the card script, Shortcut to edit the stack script, Object menu > Object Inspector

Command-Shift-Control-click (Mac OS or OS X)

Control-Shift-Right-click (Unix or Windows)

To display a controlÆs property inspector

See Also:

Shortcut to change a controlÆs properties, Shortcut to edit the card script, Edit menu > Preferences, Object menu > Object Inspector

Command-Shift-Option-hover (Mac OS or OS X)

Control-Shift-Alt-hover (Unix or Windows)

To duplicate a control

See Also:

Shortcut to select multiple controls, Shortcut to nudge the selected control, Shortcut to equalize widths of selected controls, Shortcut to equalize heights of selected controls, How to duplicate an object, Edit menu > Duplicate, Edit menu > Replicate..., clone command

Option-drag (Mac OS or OS X)

Control-drag (Unix or Windows)

To edit a controlÆs script

See Also:

Shortcut to change a controlÆs properties, Shortcut to display a controlÆs property inspector, Shortcut to edit the card script, Shortcut to edit the selected controlÆs script, Shortcut to edit the stack script, Edit menu > Preferences, Object menu > Object Script, Object menu > Card Script, Object menu > Stack Script, edit command, script property

Command-Option-hover (Mac OS or OS X)

Control-Alt-hover (Unix or Windows)

To edit the card script

See Also:

Shortcut to edit a controlÆs script, Shortcut to edit the stack script, edit command, script property

Command-Shift-C

To edit the selected controlÆs script

See Also:

Shortcut to change a controlÆs properties, Shortcut to edit a controlÆs script, edit command, script property

Command-E (Mac OS or OS X)

Control-E (Unix or Windows)

To edit the stack script

See Also:

Shortcut to edit a controlÆs script, Shortcut to edit the card script, edit command, script property

Command-Shift-S

To equalize height and width of a control

See Also:

How to change the size and position of an object, Shortcut to equalize heights of selected controls, Shortcut to equalize widths of selected controls, Object menu > Object Inspector, height property, width property

Shift key

To equalize heights of selected controls

See Also:

Shortcut to align control edges, Shortcut to equalize height and width of a control, Shortcut to select multiple controls, Shortcut to equalize widths of selected controls, Object menu > Align Selected Controls > Make Heights Equal, height property

Command-Shift-= (Mac OS or OS X)

Control-Shift-= (Unix or Windows)

To equalize widths of selected controls

See Also:

How to change the size and position of an object, Shortcut to align control edges, Shortcut to equalize height and width of a control, Shortcut to select multiple controls, Shortcut to equalize heights of selected controls, Object menu > Align Selected Controls > Make Widths Equal, width property

Command-= (Mac OS or OS X)

Control-= (Unix or Windows)

To find the selection in scripts

See Also:

(Platform-dependent)

Command-L (Mac OS or OS X)

Control-L (Unix or Windows)

To format the current handler

See Also:

(Platform-dependent

Tab key

To go to the next card

See Also:

Shortcut to review the recent cards list, Shortcut to go to the previous card, How to block or change the action of arrow keys, View menu > Go Next, go command

Right arrow

To go to the previous card

See Also:

Shortcut to review the recent cards list, Shortcut to go to the next card, How to block or change the action of arrow keys, View menu > Go Prev, go command

Left arrow

To go to the top or bottom of a field

See Also:

Shortcut to navigate in a field, `arrowKey` message, `emacsKeyBindings` property, `navigationArrows` property, `textArrows` property

Command-Up or Down arrow (Mac OS or OS X)

Control-Up or Down arrow (Unix or Windows)

To hide Revolution palettes

See Also:

Shortcut to move Revolution windows, How to hide the Toolbar, View menu > Palettes, hide command, hidePalettes property

Command-Control-Tab (Mac OS or OS X)

Control-Tab (Unix or Windows)

To look up Transcript term

See Also:

Shortcut to complete words in scripts, Shortcut to open Revolution documentation, How to quickly display the Transcript Dictionary

Control-click (Mac OS or OS X)

Right-click (Unix or Windows)

To magnify an image

See Also:

Shortcut to make an image transparent, magnify property

Command-click (Mac OS or OS X)

Control-right-click (Unix or Windows)

To make an image transparent

See Also:

Shortcut to magnify an image, import paint command

Control-click (Mac OS or OS X)

Right-click (Unix or Windows)

To move Revolution windows

See Also:

Shortcut to hide Revolution palettes, Shortcut to select controls without moving them

Alt-drag menu bar (Unix or Windows)

To navigate in a field

See Also:

Shortcut to go to the top or bottom of a field, How to block or change the action of arrow keys, arrowKey message, emacsKeyBindings property, navigationArrows property, textArrows property

arrow keys

To nudge the selected control

See Also:

Shortcut to align control edges, Shortcut to equalize height and width of a control, Shortcut to select controls without moving them, How to block or change the action of arrow keys

arrow keys

To open a stack without the development environment

See Also:

Why don't the menus appear when I open a stack?, Development menu > Suspend Development Tools, environment function, defaultMenubar property, menubar property

Command-double-click (Mac OS or OS X)

Control-double-click (Unix or Windows)

To open Revolution documentation

See Also:

Shortcut to look up Transcript term, About the Revolution documentation, How to quickly display the Transcript Dictionary, Help menu > Documentation, help message

F1 key

Help key

To remove font changes from text

See Also:

How to change the font of text, How to change the size of text, How to change the style of text, How to change the color of text, How to remove all styles from text, Text menu > Font > Use OwnerÆs Font, Text menu > Size > Use OwnerÆs Size, Text menu > Color > Use OwnerÆs Color, backgroundColor property, effective keyword, foregroundColor property, textFont property, textSize property, textStyle property

Command-; (Mac OS or OS X)

Control-; (Unix or Windows)

To reverse Select Grouped Controls

See Also:

Edit menu > Select Grouped Controls, select command, selectGroupedControls property

Command-click (Mac OS or OS X)

Control-click (Unix or Windows)

To review message box history

See Also:

Shortcut to change message box mode, Shortcut to clear the message box, Shortcut to review the recent cards list, Tools menu > Message Box, message box keyword

Up arrow key

To review the recent cards list

See Also:

Shortcut to go to the next card, Shortcut to go to the previous card, How to block or change the action of arrow keys, View menu > Go Recent, go command, recentCards property, recentNames property

Up arrow key

To save all open stacks

See Also:

File menu > Save, revLoadedStacks function, save command, stacks function

Command-Option-S (Mac OS or OS X)

Control-Alt-S (Unix or Windows)

To select controls without moving them

See Also:

Shortcut to nudge the selected control, Shortcut to select multiple controls, select command

Control-click (Mac OS or OS X)

Right-click (Unix or Windows)

To select multiple controls

See Also:

Shortcut to align control edges, Shortcut to equalize heights of selected controls, Shortcut to equalize widths of selected controls, Shortcut to select controls without moving them, select command, selectedObject function

Shift-click

To step into the next line of a handler

See Also:

Shortcut to step over the next line of a handler, Debug menu > Step Into (Script Editor)

Space

To step over the next line of a handler

See Also:

Shortcut to step into the next line of a handler, Debug menu > Step Over (Script Editor)

Option-Space (Mac OS or OS X)

Alt-Space (Unix or Windows)

To stop a running handler

See Also:

How to prevent interrupting a handler, How to stop a running handler, Why can't I interrupt a handler?, allowInterrupts property, can'tAbort property, interrupt function

Command-. (Mac OS or OS X)

Control-. (Unix or Windows)

Control-break (Unix or Windows)

To switch between Browse and Pointer tool

See Also:

Shortcut to change message box mode, Tools menu > Browse Tool, Tools menu > Pointer Tool, choose command, tool function

Command-Option-Tab (Mac OS or OS X)

Control-Alt-Tab (Unix or Windows)

