ADOBE ILLUSTRATOR® 10 JAVASCRIPT REFERENCE

Adobe Developer Support
345 Park Avenue
San Jose, CA 95110-2704
408-536-9000
FaxYI: 206-628-5737
ada@adobe.com

Europe:
PO Box 12356
Edinburgh EH11 4GJ
United Kingdom
+44.131.458.6800
Fax: +44.131.458 6801
euroADA@adobe.com

http://partners.adobe.com

Adobe

Adobe Illustrator 10 JavaScript Reference

Table of Contents

Introduction

About this reference

This reference section describes the objects and commands in Illustrator's JavaScript type library. All of the classes in the type library are presented alphabetically. The chapter concludes with a list of all of the constants in the Illustrator type library.

Each class listing includes the following:

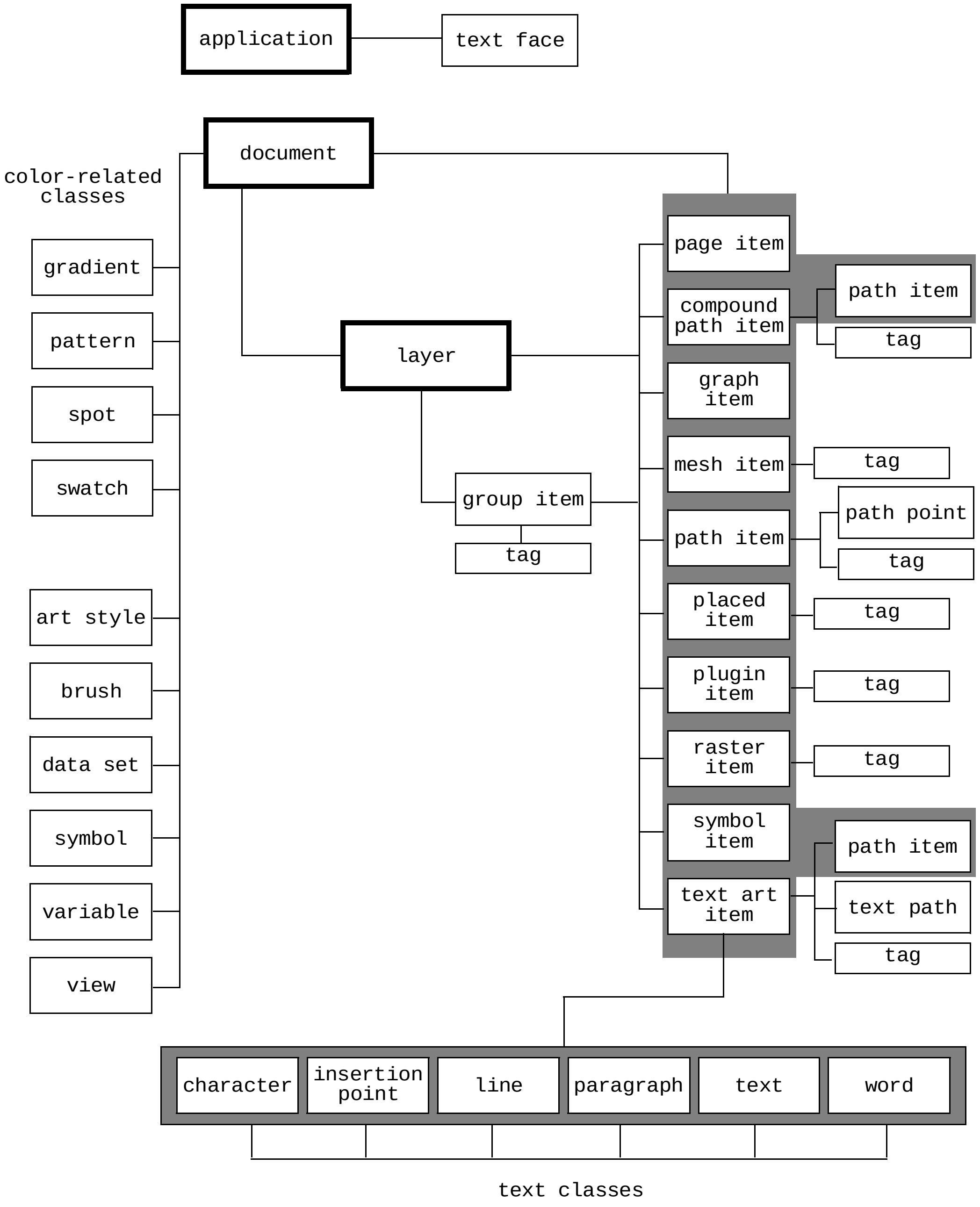
- Properties of the class, including value type, read-only status, and an explanation.
- Methods for the class. Constants and value types needed by the method are shown in bold face. Required terms are shown in plain face. All items surrounded by brackets [ ] are optional.
- Notes to explain special issues.
- Script examples.

About the script examples

Many of these examples do not show the most efficient way to construct a JavaScript statement, but they are written to be easy to read and understand. Error checking code, for example, is brief in most of the examples—the point is to show you how to address and work with the Illustrator objects. Many of the examples may be combined to make scripts with greater functionality.

Illustrator's object model

A good understanding of Illustrator's object model will improve your scripting abilities. This diagram shows the containment hierarchy of the object model, starting with the Application object. It is useful to note that all text classes as well as the `Layer` and `GroupItem` classes can contain additional objects of the same class which can, in turn, contain additional nested objects.


application
text face
document
color-related classes
gradient
pattern
spot
swatch
art style
brush
data set
symbol
variable
view
layer
group item
tag
page item
compound path item
path item
tag
graph item
mesh item
tag
path item
path point
tag
placed item
tag
plugin item
tag
raster item
tag
symbol item
path item
text art item
text path
tag
character
insertion point
line
paragraph
text
word
text classes

Referencing and Creating Objects in JavaScript

As the object model diagram shows, all objects are arranged in a hierarchy. To obtain a reference to a specific object you need to navigate the hierarchy. For example, to store a reference to the first path item in the second layer of the active document in the variable `pathRef` you would write:

```
pathRef = activeDocument.layers[1].pathItems[0];
```

Note: All array references in JavaScript are zero-based. That is, the first element of an array is index [0].

You can refer to objects in a collection by index number ([i]) or by name. For example, you can refer to the "Black" swatch as:

```
activeDocument.swatches[4]    or
activeDocument.swatches["Black"]
```

The following collection objects do not have names. You can only reference them by index.

- Characters
- GradientStops
- Paragraphs
- PathPoints
- TextLines
- Words

Since all JavaScript scripts are executed from within the Illustrator application, a specific reference to the application object is not required. For example, to assign the active document in Illustrator to the variable `frontMostDocument`, you would reference the activeDocument property of the application object as follows:

```
frontMostDocument = activeDocument;
```

There are a number of objects that cannot be obtained by using the hierarchy shown in the object model diagram. These objects must created explicitly by defining a variable, using the new object constructor, and assigning objects or values to them.

For example, to create the new `CMYKColor` object, `newCMYKColor`, you would write:

```
var newCMYKcolor = new CMYKColor();
```

These objects are:

- CMYKColor
- Color
- EPSSaveOptions
- ExportOptionsFlash
- ExportOptionsGIF
- ExportOptionsJPEG
- ExportOptionsPhotoshop
- ExportOptionsPNG8
- ExportOptionsPNG24
- ExportOptionsPS5
- ExportOptionsSVG
- File
- Folder
- GradientColor
- GrayColor
- IllustratorSaveOptions
- Matrix
- PatternColor
- PDFOpenOptions
- PDFSaveOptions
- RGBColor
- SpotColor

The following example demonstrates how to create a new SaveOptionsEPS object, assign values to it, and save it to a file named \temp\sample.eps.

```
// Create an EPS-save option object
    var fileSpec = new File("//temp/sample.eps");
    var type = ExportType.EPS
    var newSaveOptions = new SaveOptionsEPS();

// Set the options to define how the EPS document is saved by Illustrator
    newSaveOptions.embedAllFonts = true;
    newSaveOptions.compatibility = Compatibility.ILLUSTRATOR7;
    newSaveOptions.preview = EPSPreview.COLORTIFF;

// Save the active document
    activeDocument.saveAs(fileSpec, type, newSaveOptions);
```

Working with Files and Folders

Although you can use operating system dependent file and path names, the *File* and *Folder* objects use an URL-like notation. A full path starts with a slash character; each element is separated by slashes. The first element is supposed to be the volume name on a Macintosh; on Windows, the first element should be the drive name. If the path name starts with two consecutive slashes, the default volume is assumed. On Unix systems, two consecutive slashes are converted to a single slash, since Unix does not know volume names. The special path elements "." and ".." are recognized on the Macintosh and translated to the current and parent folders, respectively. The usage of the slash character as part of a file name, although a valid character on the Macintosh, is not supported. Optionally, the path can begin with the string "file://".

You should always pay attention to the case of the characters of a path name. Although the Windows and Macintosh file systems are case-insensitive, other file systems are not, like Unix file systems. If you want to write portable programs, use the correct case in your path names.

Unfortunately, the file system conventions are quite distinct for the root elements of a full path name. It is, therefore, a good idea to use relative path names instead of absolute path names if you want to write portable scripts. To illustrate how the root element of a full path name is used on different file systems, consider the following examples. The current drive is C: on Windows, and "Macintosh HD" on the Macintosh.

URL	Windows Name	Macintosh Name
/d/dir/name.ext	D:\dir\name.ext	Macintosh HD:d:dir:name.ext
//dir/name.ext	C:\dir\name.ext	Macintosh HD:dir:name.ext
/Macintosh HD/dir/name.ext	C:\Macintosh HD\dir\name.ext	Macintosh HD:dir:name.ext

If a full path name does not contain a valid drive letter in Windows, the first element is considered to be a directory on the default drive. If the first element does not match a valid volume name on the Macintosh, the first element again is treated as a folder name in the default volume's root folder. You might, therefore, end up with a folder "d" on the Mac, or a directory "Macintosh HD" on Windows.

Working with Methods

When you work with methods that have multiple parameters, you may omit optional parameters at the end of the parameter list, but you may not omit parameters in the middle of the list. If you do not wish to specify a particular parameter in the middle of the list, you must insert the string "undefined". The JavaScript interpreter will insert the default parameters wherever "undefined" is specified. For example, if you wish to rotate an object 30 degrees and have the fillGradients changed also, you would write:

```
myObject.rotate(30, undefined, undefined, true)
```

Note that you only need to specify "undefined" for the changePositions and changeFillPatterns parameters. You do not have to specify anything for the parameters that follow the changeFillGradients parameter.

Working with the Selection Object

The document `selection` object returns either an array of art objects or a string of text. To correctly handle the selection, you must first test to see if you have artwork or text. The properties of the art items in the array of artwork items can only be accessed by referencing the individual items in the array.

The following example checks the type of data contained in the selection object, assigns a name to the object, and then displays the typename and toString() values for art items or returns a message that you have selected text.

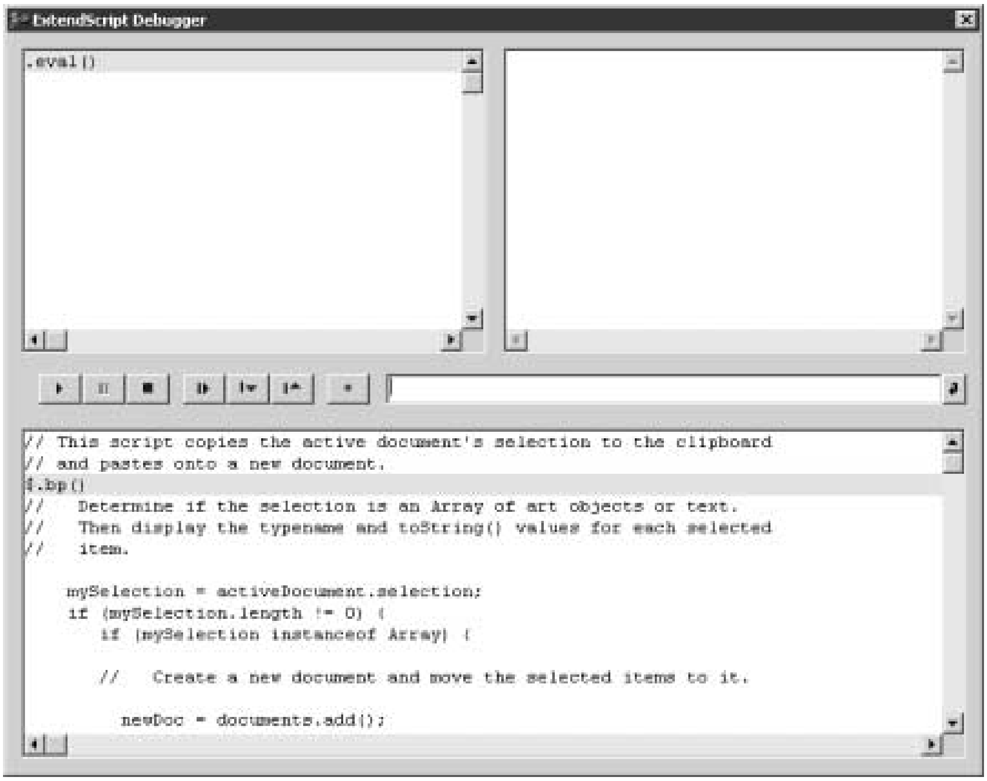
```
//   Determine if the selection is an Array of art objects or text.
//   Then display the typename and toString() values for each selected
//   item.

    mySelection = activeDocument.selection;
    if (mySelection instanceof Array) {
       msgType = "Selection items: ";
       msgString = "Selection items: ";
       for (i=0; i<mySelection.length; i++) {
           mySelection[i].name = "SelectItem " + i;
           msgType = msgType + "\nItem[" + i + "] typename is: " +
                                            mySelection[i].typename;
            msgString = msgString + "\nItem[" + i + "] toString is: " +
                                            mySelection[i].toString();
      }
     }
     else {
         msgType = "You have selected text.";
         msgString = "You have selected text.";
    }
    alert(msgType);
    alert(msgString);
```

Debugging JavaScript

To invoke the JavaScript debugger, enter the following line in your script.

$.bp()

When this statement is executed, the debugger window opens and provides three panes of information to assist you in testing your script.



```
.eval()
```

```
// This script copies the active document's selection to the clipboard
// and pastes onto a new document.
$.bp()
//   Determine if the selection is an Array of art objects or text.
//   Then display the typename and toString() values for each selected
//   item.

    mySelection = activeDocument.selection;
    if (mySelection.length != 0) {
       if (mySelection instanceof Array) {

       //   Create a new document and move the selected items to it.

         newDoc = documents.add();
```

The current stack trace appears in the upper left pane of the script debugger window. This view displays the calling hierarchy at the time of a breakpoint. Double clicking a line in this view changes the current scope enabling you to inspect scope specific data.

The rightmost panel is used for expression values.

The text field to the right of the buttons allows you to execute JavaScript expressions. The result of executing a JavaScript expression is shown in the right most panel. This can be used to look at the values of variables and objects. If you have a script that includes `var myVar = 17;`

Then entering "myVar" in the text field and pressing enter will display 17 in the rightmost panel.

The JavaScript source appears in the lower pane of the script debugger window.

Controlling Code Execution in the Script Debugger Window

Six buttons control the execution of code when the debugger is active.

Resume	This button resumes execution of the script following a breakpoint. When the script terminates, the debugger window closes automatically. Closing the debugger window manually also causes script execution to resume. This button is enabled when script execution is paused or stopped.
Pause	This button halts the currently executing script temporarily and reactivate the script debugger window. This button is enabled when a script is running.
Stop	This button stops execution of the script and generates a runtime error. This button is enabled when a script is running.
Step into	This button halts script processing after executing a single JavaScript statement in the script or after executing a single statement in any JavaScript function called by the script.
Step over	This button halts script processing after executing a single JavaScript statement in the script. If the statement calls a JavaScript function, execute the function in its entirety before stopping.
Step out	When the debugger is paused within the body of a JavaScript function, this button resumes script execution until the function returns. When paused outside the body of a function, clicking this button resumes script execution until the script terminates.
●	This button is not implemented at this time.

1 JavaScript Reference

Application

The Adobe Illustrator application object, which contains all other Illustrator objects.

Properties

Property	R/O	Value type	What it is
activeDocument		Document object	The active (frontmost) document in Illustrator.
browserAvailable	R/O	Boolean	Is a web browser available?
documents	R/O	Documents collection object	The documents in the application.
freeMemory	R/O	Number	The amount of unused memory (in bytes) within the Adobe Illustrator partition.
name	R/O	String	The application's name (not related to the filename of the application file).
parent	R/O	Layer object or GroupItem object	The parent of this object.
path	R/O	File object	The file path to the application.
preferences		Preference object	The preference settings for Illustrator.
scriptingVersion	R/O	String	The version of the Scripting plugin.
selection		Array (of objects)	All of the currently selected objects in the active (frontmost) document. See note for more information.
textFaces	R/O	TextFaces collection object	The text faces (fonts) available to the application.

Property	R/O	Value type	What it is
typename	R/O	String	Returns the name of the referenced object.
userInteractionLevel		UserInteractionLevel constant	Should alerts be displayed?
version	R/O	String	The version of the Adobe Illustrator application.
visible	R/O	Boolean	Is the application visible?

Methods

Method	Parameter type	Returns	What it does
alert(msgText)	String	Nothing	Displays a message box on the screen.
beep()	none	Nothing	Sounds a beep tone.
concatenateMatrix(matrix, secondMatrix)	 Matrix object Matrix object	Matrix object	Joins two matrices together.
concatenateRotationMatrix(matrix, angle)	 Matrix object number	Matrix object	Joins a rotation translation to a transformation matrix.
concatenateScaleMatrix(matrix [,scaleX] [,scaleY])	 object number number	Matrix object	Concatenates a scale translation to a transformation matrix.
concatenateTranslationMatrix(matrix [,deltaX] [,deltaY])	 Matrix object number number	Matrix object	Joins a translation to a transformation matrix.
confirm(confirmMsg)	String	Boolean	Displays a confirmation message box and returns the users response (Yes/ No).
getIdentityMatrix()	none	Matrix object	Returns an identity matrix.
getRotationMatrix([angle])	number	Matrix object	Returns a transformation matrix containing a single rotation.
getScaleMatrix([scaleX] [, scaleY])	 number number	Matrix object	Returns a transformation matrix containing a single scale.
getTranslationMatrix([deltaX] [, deltaY])	 number number	Matrix object	Returns a transformation matrix containing a single translation.

Method	Parameter type	Returns	What it does
invertMatrix(matrix)	Matrix object	Matrix object	Inverts a matrix.
isEqualMatrix(matrix, secondMatrix)	 Matrix object Matrix object	Boolean	Are two matrices equal.?
isSingularMatrix(Matrix)	Matrix object	Boolean	Checks whether a matrix is singular and cannot be inverted.
open(fileSpec [, documentColorSpace] [,options])	 File object ColorSpace constant PDFOpenOptions object (for PDF files only)	Document object	Opens the file specified by the string with the specified color space and options.
quit		Nothing	Quits Illustrator. Note that if the clipboard contains data, Illustrator may show a dialog prompting the user to save the data for other applications.
redraw		Nothing	Allow Illustrator to redraw all its windows.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

In this script, we use the application property `activeDocument` to duplicate the current document's selection before moving it into a new document. This script also demonstrates how to create the new document.

```
// This script duplicates the selection from the active document
// and places it in a new document.

// To invoke the JavaScript debugger, remove the comment delimiters
// from the next line
// $.bp()

//   Determine if the selection is an Array of art objects or text.
//   Then display the typename and toString() values for each selected
//   item.

     mySelection = activeDocument.selection;
     if (mySelection.length != 0)
     {
        if (mySelection instanceof Array)
        {
           // Create a new document and move the selected items to it.

           newDoc = documents.add();
           for (i=0; i<mySelection.length; i++)
           {
               newItem = mySelection[i].duplicate();
               newItem.moveToBeginning(newDoc);
           }
        }
           else
           {
              newDoc = documents.add();
              newItem = mySelection.parent.duplicate();
              newItem.moveToBeginning(newDoc);
           }
     }
        else
        {
           alert("Please select one or more art objects");
        }
```

ArtStyle

An art style. Each art style defines a set of appearance attributes that you can apply nondestructively to page items. Art styles are contained in documents.

Properties

Property	R/O	Value type	What it is
name	R/O	String	The art style name.
parent	R/O	Document object	The document that contains this art style.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type:	Returns	What it does
applyTo(artItem)	ArtItem object	Nothing	Applies the art style to a specific art object.
remove()	none	Nothing	Removes the referenced item from the document.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Illustrator’s `artStyle` object represents an art style as defined in the Illustrator document. Additional art styles may be created by the user within Illustrator. Art styles cannot be created by a script.

Example

This example duplicates and groups the current selection, applying the third art style in the document to the items in the group

```
// This script duplicates the selection and places it into a new group
// then applies an art style to the new group’s items

if (documents.length > 0)
{
   selectedItems = activeDocument.selection;

   newGroup = activeDocument.groupItems.add();
   newGroup.name = "TheNewGroup";
   newGroup.moveToEnd(activeDocument);

   endIndex = selectedItems.length;

   for (i = endIndex-1; i >= 0; i--)
   {
      pageObject = selectedItems[i];
      pageItemType = pageObject.typename;

      if (pageItemType == "PathItem")
      {
         newPath = pageObject.duplicate();
         newPath.moveToBeginning(newGroup);
      }
   }

   for (i=0; i < newGroup.pageItems.length; i++)
   {
      newGroup.pageItems[i].artStyle = activeDocument.artStyles[2];
   }
}
```

ArtStyles

A collection of art styles in a document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of artstyles in the document.
parent	R/O	Object	The document that contains this artstyles collection
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type:	Returns	What it does
removeAll()	none	Nothing	Removes all objects from the referenced collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Illustrator’s `ArtStyle` object represents an art style as defined in the Illustrator document. Additional art styles may be created by the user within Illustrator. Art styles cannot be created by a script.

Example

This script displays the total number of art styles available in the current document.

```
// This script finds the number of art styles in the active document

if (documents.length > 0)
{
   numberOfStyles = activeDocument.artStyles.length;
   alert("There are " + numberOfStyles + " art styles in the active 
document.");
}
```

Brush

A brush in an Illustrator document. Brushes are contained in documents.

Properties

Property	R/O	Value type	What it is
name	R/O	String	The brush name.
parent	R/O	Document object	The document that contains this brush.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
applyTo(artItem)		Nothing	Applies the brush to a specific art object.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Additional brushes may be created by the user within Illustrator. You can access Illustrator's brushes within a script, but you cannot create them.

Example

This example duplicates and groups the current selection, applying the third art style in the document to the items in the group.

```
// This script duplicates and groups the current selection
// applies a brush to the new group’s items

if (documents.length > 0)
{
   mySelections = activeDocument.selection;

    if (mySelections instanceof Array)
   {
        newGroup = activeDocument.groupItems.add();
       len = activeDocument.selection.length;

        for (i=0; i < len; i++)
       {
            newItem = mySelections[i].duplicate();
            newItem.moveToBeginning(newGroup);
        }

       theBrush = activeDocument.brushes[2];
       theBrush.applyTo(newGroup);
    }
}
```

Brushes

A collection of brushes in a document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The document that contains this brushes collection
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Illustrator's `Brush` object represents a brush as defined in the Illustrator document. You can create additional brushes in Illustrator, but you cannot create them in a script.

Example

This script displays the total number of available brushes in the current document.

```
// This script counts all brushes in the active document and
// displays the total

if (documents.length > 0)
{
   numberOfBrushes = activeDocument.brushes.length;
   alert ("There are " + numberOfBrushes + " brushes in the active
document.");
}
```

Character

A single character of text in the contents of a text art item.

Properties

Property	**R/O**	**Value type**	**What it is**
autoKerning		Boolean	Should a font’s built-in kerning information be used?
baselineShift		Number	Baseline offset of text.
clipping	R/O	Boolean	Is there a clipping path associated with the text art item containing this character?
contents		String	The text contained in the text range.
direction		CharacterDirection constant	The orientation of the characters in a vertical text block.
evenodd		Boolean	Should the even-odd rule be used to determine insideness?
fillColor		Color	Fill color of text
filled		Boolean	Should the text be filled?
fillOverprint		Boolean	Should the art beneath the text be overprinted?
font		String	The text face of the text.
kerning		Number	The spacing between two characters in milli-ms.
leading		Number	The vertical leading of the text.
length	R/O	Number	The number of character in the text.
note	R/O	String	The note associated with this text.
offset	R/O	Number	Offset of selected text in text range (in characters).
orientation	R/O	TextOrientation constant	The orientation of the text. Use the TextPath class to alter this property.
paragraph	R/O	Paragraph object	The paragraph containing the character.
parent	R/O	TextArtItem object	The parent of this object.
resolution	R/O	Number	The resolution of the object (in dots per inch).

Property	R/O	Value type	What it is
scaling		Array (of 2 numbers)	The character scaling supplied as a point with the first coordinate as horizontal scale and the second coordinate as vertical scale, where 100.0 is 100%.
size		Number	Font size of text.
strokeCap		StrokeCap constant	The type of line capping.
strokeColor		Color object	The stroke color for the path.
stroked		Boolean	Should the path be stroked?
strokeDashes		Array	Dash lengths. Set to an empty array for a solid line.
strokeDashOffset		Number	The default distance into the dash pattern at which the pattern should be started.
strokeJoin		StrokeJoin constant	Type of joints for the path.
strokeMiterLimit		Number	Are joins mitered (pointed) or beveled (squared-off)?
strokeOverprint		Boolean	Will art beneath a stroked object be overprinted?
strokeWidth		Number	Width of stroke.
textLine	R/O	TextLine object	The line of text containing the character.
textPath	R/O	TextPath object	A reference to the text path associated with the text art item containing this text.
tracking		Number	The spacing between multiple characters.
typename	R/O	String	Returns the name of the referenced object.
word	R/O	TextRange object	The word containing this character.

Methods

Method	Parameter type	Returns	What it does
remove()	none	Nothing	Removes the referenced item from the document.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

The text contained within text art items in Illustrator can be accessed using the `Character`, `Word`, `TextLine`, `Paragraph` and `TextRange` objects. The properties and valid commands for these objects are similar, but not identical. For example, `Character` has a `kerning` property, but other text objects do not.

Example

This example demonstrates how to use character properties to create unique effects from a script.

```
// This script distorts all characters in all text art items
// by incrementally modifying the vertical scaling of each chararacter
// to give the effect of stretching words out.

var pi = 3.1415;

if (documents.length != 0)
{
   for (i=0; i < activeDocument.textArtItems.length; i++)
   {
      theTextArt = activeDocument.textArtItems[i];
      textArtRange = theTextArt.textRange();
      characterCount = textArtRange.characters.length;
      index = 0;
      for (j=0; j < characterCount; j++)
      {
         textCharacter = textArtRange.characters[j];
         with (Math)
         {
            verticalScale = sin(pi * index/characterCount) * 200 + 200;
         }

         horizScale = verticalScale * .75;
         textCharacter.scaling = Array(horizScale, verticalScale);
         index++;
      }
   }
}
```

Characters

A collection of characters.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of characters in the collection.
parent	R/O	Object	The text art item that contains this character.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	Character object	Add a character after the last character in the current collection.
addBefore()	none	Nothing	Adds a character before the current paragraph selection or insertion point.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This script displays the total number of characters contained in all of the text art items in the current document.

```
// This script counts all characters, including whitespace,
// in the active document and returns the total

if (documents.length > 0)
{
   numChar = 0;
   for (i=0; i < activeDocument.textArtItems.length; i++)
   {
      textArtRange = activeDocument.textArtItems[i].contents;
      numChar = numChar + textArtRange.length;
   }

   alert("There are " + numChar + " characters, including whitespace, in
the document.");
}
```

CMYKColor

A CMYK color specification, used in conjunction with the `CMYK` property of the `color` object.

Properties

Property	R/O	Value type	What it is
black		Number	The black color value as a value in the range 0.0 - 100.0.
cyan		Number	The cyan color value as a value in the range 0.0 - 100.0.
magenta		Number	The magenta color value as a value in the range 0.0 - 100.0.
typename	R/O	String	Returns the name of the referenced object.
yellow		Number	The yellow color value as a value in the range 0.0 - 100.0.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

If the color space of a document is `RGB` and you specify the color value for a page item in that document using `CMYK`, Illustrator will translate the CMYK color specification into an RGB color specification. The same thing happens if the document's color space is `CMYK` and you specify colors using `RGB`. Since this translation can lose information, you should specify colors using the class that matches the document's actual color space.

Example

This script sets the fill color for the frontmost PathItem in the active document to a light shade of purple.

```
// This script sets the fill color of the frontmost path item in
// the current document to a light purple CMYK color

if (documents.length > 0 && activeDocument.pathItems.length > 0)
{
   frontPath = activeDocument.pathItems[0];

   // Set color values for the CMYK objects
   // Then wrap the color in a standard color object

    newCMYKColor = new CMYKColor();

    newCMYKColor.black = 0;
    newCMYKColor.cyan = 30.4;
    newCMYKColor.magenta = 32;
    newCMYKColor.yellow = 0;

    var newColor = new Color();
    newColor.cmyk = newCMYKColor;

    frontPath.filled = true;
    frontPath.fillColor = newColor;
}
```

Color

A general color specification that includes a color space specification as well as a specific color specification for the color space selected.

Properties

Property	R/O	Value type	What it is
cmyk		CMYKColor object	A CMYK color specification.
color	R/O	Color constant	The color space for this color. Any color specification included in the color info specification must correspond to the color space, i.e. if model is `CMYK`, the color specification object included must be a CMYKColor object.
gradient		GradientColor object	A gradient color specification.
gray		GrayColor object	A gray color specification.
pattern		PatternColor object	A pattern color specification.
rgb		RGBColor object	A RGB color specification.
spot		Spot object	A spot color specification.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

All colors are specified in Illustrator using `color`, except the `color` property of layers which is specified directly as an RGB Color specification using `RGBColor`. To set a color, you do not need to specify the `Color` property. Illustrator will imply the color space based on the other properties included.

Example

The following script examines the color of the frontmost PathItem in the current document. Note that a document using the CMYK color space will never return an RGB color. A document using the RGB color space will never return a CMYK color.

```
// This script sets the fill color of the frontmost path item in
// the current document to a light purple CMYK color

if (documents.length > 0 && activeDocument.pathItems.length > 0)
{
   newCMYKColor = new CMYKColor();
   newColor = new Color();

   // Get a reference to the frontmost path in the document
   frontPath = activeDocument.pathItems[0];

   // Set color values for the CMYK objects
   // Then wrap the color in a standard color object

   newCMYKColor.black = 0;
   newCMYKColor.cyan = 30.4;
   newCMYKColor.magenta = 32;
   newCMYKColor.yellow = 0;

   newColor.cmyk = newCMYKColor;

   frontPath.filled = true;
   frontPath.fillColor = newColor;
}
```

CompoundPathItem

A compound path. Compound paths are objects composed of multiple intersecting paths, resulting in transparent interior spaces where the original paths overlapped.

Properties

Property	R/O	Value type	What it is
artworkKnockout		KnockoutState constant	Is this object used to create a knockout? If so, what kind of knockout?
blendingMode		BlendModes constant	The mode used when compositing an object.
controlBounds	R/O	Array (of 4 numbers)	The bounds of the object including stroke width and controls.
editable	R/O	Boolean	Is this item editable?
geometricBounds	R/O	Array (of 4 numbers)	The bounds of the object excluding stroke width.
height		Number	The height of the compound path item excluding stroke width.
hidden		Boolean	Is this compound path item hidden?
isIsolated		Boolean	Is this object isolated?
layer	R/O	Layer object	The layer to which this compound path item belongs.
left		Number	The position of the left side of the item.
locked		Boolean	Is this compound path item locked?
name		String	The name of this compound path item.
opacity		Number	The opacity of the object . The value is between 0.0 and 100.0.
parent	R/O	Layer object or GroupItem object	The parent of this object.
pathItems	R/O	PathItems collection object	The path art items in this compound path.
position		Array (of 2 numbers)	The position of the top left corner of the compound path item excluding stroke width.
selected		Boolean	Is this compound path item selected?

Property	R/O	Value type	What it is
sliced		Boolean	Is the item sliced? Default: false
tags	R/O	Tags collection object	The tags contained in this object.
top		Number	The position of the top of the item.
typename	R/O	String	Returns the name of the referenced object.
url		String	The value of the Adobe URL tag assigned to this compound path item.
visibilityVariable		Variable object	The visibility variable bound to the item.
visibleBounds	R/O	Array (of 4 numbers)	The visible bounds of the compound path item including stroke width.
width		Number	The width of the compound path item excluding stroke width.
zOrderPosition	R/O	Number	The position of this art object within the stacking order of the group or layer (Parent) that contains the art object.

Methods

Method	Parameter type	Returns	What it does
duplicate()	none	Item	Creates a duplicate of the selected item.
moveAfter(destination)	object	Nothing	Moves the item behind the specified object.
moveBefore(destination)	object	Nothing	Moves the item in front of the specified object.
moveToBeginning(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the front of the specified container.
moveToEnd(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the end of the specified container.
remove()	none	Nothing	Removes the referenced item from the document.

Method	Parameter type	Returns	What it does
resize(scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])	 number number boolean boolean boolean boolean boolean Transformation constant	Nothing	Scales the art object where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%.
rotate(angle [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,rotateAbout])	 number boolean boolean boolean boolean Transformation constant	Nothing	Rotates the art object relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.
transform transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])	 matrix object boolean boolean boolean boolean number Transformation constant	Nothing	Transforms the art object by applying a transformation matrix.
translate([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])	 number number boolean boolean boolean boolean	Nothing	Repositions the art object relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
zOrder(ZOrderMethod)	ZOrderMethod constant	Nothing	Arranges the art object’s position in the stacking order of the group or layer (parent) of this object.

Notes

Paths contained within a compound path or group in a document will be returned as individual paths when a script asks for the paths contained in the document. However, paths contained in a compound path or group will not be returned when a script asks for the paths in a layer that contains the compound path or group.

All paths inside of a compound path share property values. Therefore, if you set the value of a property of any one of the paths in the compound path, all other path's matching property will be updated to the new value.

The `PathItems` property provides access to the paths that make up the compound path.

Example 1

This example demonstrates how to select all of the paths in a document that are not part of a compound path or a group by testing the type of the `Parent` property with a function.

```
// This script selects all paths not part of a compound path.
//$.bp();

if (documents.length > 0)
{
   count = 0;

   // clear the old selection
   activeDocument.selection = null;

   if (activeDocument.pathItems.length > 0)
   {
      thePaths = activeDocument.pathItems;
      numPaths = thePaths.length;

      for (i = 0; i < numPaths; i++)
      {
         pathArt = thePaths[i];
         if (pathArt.parent.typename != "CompoundPathItem")
         {
            pathArt.selected = true;
            count++;
         }
      }
   }
}

alert("Number of pathItems selected =  " + count);
```

Example 2

This example demonstrates how to create a new compound path containing 3 path items. The example then modifies the stroke of the paths in the compound path. Note that when you modify the properties of a PathItem inside a compound path you affect all paths contained in the compound path. The example also shows how to access swatches in a document by name.

```
// This script creates a CoumpoundPath containing 3 PathItems
// It then sets the width and the color of the stroke
// Note that when you modify a path in a compound path you affect
// all paths in the compound path

if (documents.length > 0)
{
   frontDocument = activeDocument;
   activeLayer = frontDocument.activeLayer;
   newCompoundPath = activeLayer.compoundPathItems.add();

   // Create the path items;
   newPath = newCompoundPath.pathItems.add();
   newPath.setEntirePath(Array(Array(30, 50), Array(30, 100)));

   newPath = newCompoundPath.pathItems.add();
   newPath.setEntirePath(Array(Array(40, 100), Array(100, 100)));

   newPath = newCompoundPath.pathItems.add();
   newPath.setEntirePath(Array(Array(100, 110), Array(100, 300)));

   // Set the gradient of the compound path
   newPath.stroked = true;
   newPath.strokeWidth = 3.5;
   newPath.strokeColor = frontDocument.swatches["Orange M=50 Y=100"].color;
}
```

CompoundPathItems

A collection of compound paths.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this collection. Either a Layer or a GroupItem.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	CompountPathItem object	Creates a new CompoundPathItem.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This example displays the total number of compound paths contained in the first layer of the current document.

```
// This script counts all compound paths in layer 1 of current document

if (documents.length > 0)
{
   numCompoundPaths = activeDocument.layers[0].compoundPathItems.length;
   alert("There are " + numCompoundPaths + " compound paths in the active
document.");
}
```

Dataset

A set of data used for dynamic publishing.

Properties

Property	R/O	Value type	What is it
name		String	Then name of the dataset.
parent	R/O	Document object	The name of the object that contains this dataset.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
display()	none	Nothing	Displays the dataset.
remove()	none	Nothing	Removes the referenced item from the document.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.
update()	none	Nothing	Updates the dataset.

Notes

A dataset allows you to collect a number of variables and their dynamic data into one object. You must have at least one variable bound to an art object in order to create a dataset. See the class definition for `variable` in this chapter for more information.

Datasets

A collection of datasets.

Properties

Property	R/O	Value type	What is it
length	R/O	Number	The number of datasets in the collection
parent	R/O	Document object	The name of the object that contains this dataset.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	Dataset object	Creates a new dataset object.
removeAll()	none	Nothing	Removes all datasets from the collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Document

An Illustrator document. Documents are contained in the `Application` object.

Properties

Property	R/O	Value type	What it is
activeDataset		Dataset object	The currently opened dataset.
activeLayer		Layer object	The active layer in the document.
activeView	R/O	View object	The document's current view.
artStyles	R/O	ArtStyles collection object	The art styles contained in the document.
brushes	R/O	Brushes collection object	The brushes contained in the document.
compoundPathItems	R/O	CompoundPathItems collection object	The compound path items contained in the document.
cropBox		Array (of 4 numbers)	The boundary of the document's cropping box for output. A document does not have a default CropBox. In order to read this property you have to set the CropBox first.
cropStyle		CropOptions constant	The style of the document's cropping box.
dataSets		Datasets collection object	The datasets contained in the document.
defaultFillColor		Color object	The color to fill new paths if default filled is `true`.
defaultFilled		Boolean	Should a new path be filled?
defaultFillOverprint		Boolean	Will art beneath a filled object be overprinted by default?
defaultStrokeCap		StrokeCap constant	Default type of line capping for paths created.
defaultStrokeColor		Color object	The stroke color for new paths if default stroked is `true`.
defaultStroked		Boolean	Should a new path be stroked?
defaultStrokeDashes		Array (of numbers)	Default lengths for dashes and gaps in dashed lines, starting with the first dash length, followed by the first gap length, and so on. Set to an empty Array for solid line.

Property	R/O	Value type	What it is
defaultStrokeDashOffset		Number	The default distance into the dash pattern at which the pattern should be started for new paths.
defaultStrokeJoin		StrokeJoin constant	Default type of joints in new paths.
defaultStrokeMiterLimit		Number	Specifies when a join is mitered (pointed) or beveled (squared-off) by default, when default stroke join is set to mitered.
defaultStrokeOverprint		Boolean	Will art beneath a stroked object be overprinted by default?
defaultStrokeWidth		Number	Default width of stroke for new paths.
documentColorSpace	R/O	DocumentColorSpace constant	The color specification system to use for this document's color space.
fullName	R/O	String	The file associated with the document, which includes the complete path to the file.
geometricBounds	R/O	Array (of 4 numbers)	The bounds of the illustration excluding the stroke width of any objects in the document.
gradients	R/O	Gradients collection object	The gradients contained in the document.
groupItems	R/O	GroupItems collection object	The group items contained in the document.
height	R/O	Number	The height of the document.
layers	R/O	Layers collection object	The layers contained in the document.
meshItems	R/O	MeshItems collection object	The mesh art items contained in the document.
name	R/O	String	The document's name (not the complete file path to the document).
outputResolution		Number	The current output resolution for the document in dots per inch (dpi).
pageItems	R/O	PageItems collection object	The page items (contains all art object classes) contained in the document.

Property	R/O	Value type	What it is
pageOrigin		Array (of 2 numbers)	The zero-point of the page in the document without margins, relative to the overall height and width.
parent	R/O	Application object	The application that contains this document.
path	R/O	String	The file associated with the document, which includes the complete path to the file.
pathItems	R/O	PathItems collection object	The path items contained in this document.
patterns	R/O	Patterns collection object	The patterns contained in this document.
placedItems	R/O	PlacedItems collection object	The placed items contained in this document.
pluginItems	R/O	PluginItems collection object	The plugin items contained in this document.
printTiles	R/O	Boolean	Does this document print as tiled output?
rasterItems	R/O	RasterItems collection object	The raster items contained in this document.
rulerOrigin		Array (of 2 numbers)	The zero-point of the rulers in the document relative to the bottom left of the document.
rulerUnits	R/O	RulerUnits constant	The default measurement units for the rulers in the document.
saved		Boolean	False if the document has never been saved or if the document has been changed since last time it was saved.
selection		Array (of objects)	The array of references to the objects in this document's current selection.
showPlacedImages		Boolean	Are placed images displayed in the document?
splitLongPaths		Boolean	Are long paths to be split when printing?
spots	R/O	Spots collection object	The spot colors contained in this document.
swatches	R/O	Swatches collection object	The swatches contained in this document.

Property	R/O	Value type	What it is
symbolItems		SymbolItems collection object	The ArtItems in the document linked to symbols.
symbols		Symbols collection object	The collection of symbols in the document.
tags	R/O	Tags collection object	The tags contained in this document.
textArtItems	R/O	TextArtItems collection object	The text art items contained in this document.
tileFullPages	R/O	Boolean	Should full pages be tiled when printing this document?
typename	R/O	String	Returns the name of the referenced object.
useDefaultScreen	R/O	Boolean	Should the printer's default screen be used when printing this document?
views	R/O	Views collection object	The views contained in this document.
visibleBounds	R/O	Array (of 4 numbers)	The visible bounds of the document, including stroke width of any objects in the illustration.
width	R/O	Number	The width of this document.

Methods

Method	Parameter type:	Returns	What it does
activate()	none	Nothing	Bring the first window associated with the document to the front.
close([saveOptions])	SaveOptions object	Nothing	Closes a document using specified SaveOptions.
exportFile(fileSpec, exportFormat [,ExportOptionsGIF || ExportOptionsJPEG || ExportOptionsPNG24 || ExportOptionsPNG8 || ExportOptionsPhotoshop || ExportOptionsSVG])	File object ExportType constant object (appropriate export options object for the type of file being exported)	Nothing	Exports the document to the specified file using one of the export file formats.

Method	**Parameter type:**	**Returns**	**What it does**
exportVariables(fileSpec)	File object	Nothing	Save datasets into an XML library. The datasets contain variables and their associated dynamic data.
importVariables(fileSpec)	File object	Nothing	Import a library containing datasets, variables, and their associated dynamic data. Importing variables overwrites existing variables and datasets.
print([showDialog])	boolean	Nothing	Prints the document.
save()	none	Nothing	Saves the document in it current location.
saveAs([fileSpec] [,EPSSaveOptions || IllustratorSaveOptions || PDFSaveOptions])	 File object object object object	Nothing	Saves the document in the specified file as an Illustrator, EPS, or PDF file.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

In Illustrator, your `selection` can be accessed as well as modified. The selection contains `null` when nothing is selected. To deselect all objects in the current document, set `activeDocument.selection` to `null`, as the following example shows.

```
var docRef = activeDocument;
docRef.selection = null;
```

The frontmost document can be referred to as either `activeDocument` or `documents[0]`.

Illustrator’s default document settings—those properties starting with the word “default”—are global settings that affect the current document. Be sure to modify these default properties only when a document is open. Note that if you set default properties to desired values before creating new objects, you can streamline your scripts, eliminating the need to specify specific properties such as `fillColor` and `stroked` that have default properties.

A document's color space, height, and width can only be set when the document is created. After you create the document, these properties cannot be changed.

When you export a document with the `exportFile()` method, the appropriate file extension is appended for the following file types.

- GIF files append .gif
- JPEG files append .jpg
- PNG24 and files append .png
- SVG files append .svg

When you export Photoshop documents, the file extension is not automatically appended. You must include the file extension (.psd) in the file specification.

If you close the document, you should set your document reference to `null` to prevent your script from accidentally trying to access closed documents (see example 12.1).

A reference to an insertion point is returned when there is an active insertion point in the contents of a text art item. Similarly, a reference to a range of text is returned when characters are selected in the contents of a text art item.

If the `open` method is called to open a pre-Illustrator 9 document that contains both RGB and CMYK colors and the `documentColorSpace` parameter is not supplied, Illustrator will display a dialog to the user. When the `documentColorSpace` parameter is supplied and Illustrator encounters documents containing both color spaces, the document will be opened without a dialog and all colors will be converted to the specified color space.

Example 1

The following example shows how to make sure a document is open before setting any of the default properties.

```
// close the active document without saving changes.

if (documents.length > 0)
{
   aiDocument = activeDocument;
   aiDocument.close(SaveOptions.DONOTSAVECHANGES);
   aiDocument = null;
}
```

Example 2

This example demonstrates how to create a new document with specific default properties.

```
// This script creates a document if none exist
// and then sets fill and stroke defaults

if (documents.length == 0)
{
   frontDocument = documents.add();
}
else
{
   frontDocument = documents[0];
}

frontDocument.defaultFilled = true;
frontDocument.defaultStroked = true;
```

Documents

A collection of documents.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add([documentColorSpace] [,Width] [,Height])	DocumentColorSpace constant number number	Document object	Creates a new document using optional parameters and returns a reference to the new document.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This examples demonstrates how to create a new document with a specific color space.

```
// This script creates a document with RGB color space

documents.add(DocumentColorSpace.RGB);
```

EPSSaveOptions

Options which may be supplied when saving a document as an Illustrator EPS file. See the `document.saveAs` method for additional details.

Properties

Property	R/O	Value type	What it is
cmykPostScript		Boolean	Use CMYK PostScript?
compatibility		Compatibility constant	Specifies the version of the EPS file format to save.
embedAllFonts		Boolean	Include fonts used in the EPS file?
embedLinkedFiles		Boolean	Are linked image files to be included in the saved document?
flattenOuput		OutputFlattening constant	How should transparency be flattened for file formats older than Illustrator 9?
includeDocumentThumbnails		Boolean	Include thumbnail image of the EPS artwork?
japaneseFileFormat		Boolean	Save file using Japanese version of file format?
postScript		PostScriptLevel constant	Specifies the PostScript level to use when saving the file.
preview		EPSPreview constant	Specifies the format for the EPS preview image.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

`EPSSaveOptions` can only be supplied in conjunction with the `document.saveAs()` method.

It is not necessary to specify values for all properties. Default values will be provided for any properties not specified.

Example

This example demonstrates how to save the current document as an Illustrator 8-compatible EPS file using CMYK PostScript with all fonts embedded.

```
// This script saves the current document as an EPS with specific options

if (documents.length > 0)
{
   newSaveOptions = new EPSSaveOptions();
   newFile = new File("//temp/sample.eps");
   frontDocument = activeDocument;

    newSaveOptions.cmykPostScript = true;
    newSaveOptions.compatibility = Compatibility.ILLUSTRATOR8;
    newSaveOptions.embedAllFonts = true;

    frontDocument.saveAs(newFile, newSaveOptions);
}
```

ExportOptionsFlash

Options which may be supplied when exporting a document as a GIF file. See the `exportFile` method for additional details.

Properties

Property	R/O	Value type	What it is
artBoardClipping		Boolean	Should the exported image be clipped to the art board? The default value is `false`.
curveQuality		Number	The amount of curve information that should be presented. Default: 7
exportStyle		FlashExportStyle constant	The style in which the exported data should be created in Flash. Default: ASFLASHFILE
frameRate		Integer (1 - 120)	The display rate in frames per second. Default: 12
generateHTML		Boolean	Should the image be exported as an HTML file. Default: true
imageFormat		FlashImageFormat constant	How should the image in the exported Flash file be compressed. Default: LOSSLESS
jpegMethod		FlashJPEGMethod constant	The JPEG method to use.
jpegQuality		Integer (0 - 3)	Level of compression to use. Default: 3
looping		Boolean	Should the Flash file be set to loop when run. Default: false
readOnly		Boolean	Export as read-only file. Default: false
replacing		SaveOptions constant	If a file with the same name already exists, should it be replaced. Default: PROMPTTOSAVECHANGES
resolution		Integer (72 - 2400)	Pixels per inch. Default: 72
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

When you export a document as a Flash file with the `exportFile()` method, the file extension .html is appended automatically. You should not include any file extension in the file specification.

Example

This example demonstrates how to export the current document as a Flash file.

```
// This script exports the current document as a Flash file with
// resolution 150 dots per inch.

    var docRef = activeDocument;
    var exportOptions = new ExportOptionsFlash();
    var type = ExportType.FLASH;
    var fileSpec = new File("//temp/sample.swf");

    if (documents.length > 0)
    {
        exportOptions.resolution = 150;

        activeDocument.exportFile(fileSpec, type, exportOptions);
    }
```

ExportOptionsGIF

Options which may be supplied when exporting a document as a GIF file. See the `exportFile` method for additional details.

Properties

Property	R/O	Value type	What it is
antiAliasing		Boolean	Should the exported image be anti-aliased? Default: true
artBoardClipping		Boolean	Should the exported image be clipped to the art board? Default: false
colorCount		Number	The number of colors in the exported image's color table. Acceptable values range from 2 to 256. Default: 128.
colorDither		ColorDitherMethod constant	The method used to dither colors in the exported image. Default: DIFFUSIONDITHER
colorReduction		ColorReductionMethod constant	The method used to reduce the number of colors in the exported image. Default: `SELECTIVE`
ditherPercent		Number	How much should the colors of the exported image be dithered, where 100.0 is 100%.
horizontalScale		Number	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0.
infoLossPercent		Number	The level of information loss allowed during compression, where 100.0 is 100%.
interlaced		Boolean	Should the exported image be interlaced? Default: false
matte		Boolean	Should the art board be matted with a color? The default value is `true`.

Property	R/O	Value type	What it is
matteColor		RGBColor object	The color to use when matting the art board. Default: WHITE
saveAsHTML		Boolean	Should the exported image be saved with an accompanying HTML file? The default value is `false`.
transparency		Boolean	Should the exported image use transparency? The default value is `true`.
typename	R/O	String	Returns the name of the referenced object.
verticalScale		Number	The vertical scaling factor to apply to the exported image, where 100.0 is 100%. The default value is 100.0.
webSnap		Number	How much should the color table be changed to match the web palette, where 100 is maximum. The default value is 0.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

`ExportOptionsGIF` can only be supplied in conjunction with the `exportFile` method.

It is not necessary to specify values for all properties. Default values will be provided for any properties not specified.

Example

This example demonstrates how to export the current document as a GIF.

```
// This script saves the current document as a GIF file with
// specific options

   var exportOptions = new ExportOptionsGIF();
   var type = ExportType.GIF;
   var fileSpec = new File("//temp/sample.gif");
   var docRef = activeDocument;

   if (documents.length > 0)
   {
      exportOptions.antiAliasing = false;
      exportOptions.colorCount = 64;
      exportOptions.colorDither = ColorDitherMethod.DIFFUSION;

      docRef.exportFile(fileSpec, type, exportOptions);
   }
```

ExportOptionsJPEG

Options which may be supplied when exporting a document as a JPEG file. See the `exportFile` method for additional details.

Properties

Property	R/O	Value type	What it is
antiAliasing		Boolean	Should the exported image be anti-aliased? The default value is `true`.
artBoardClipping		Boolean	Should the exported image be clipped to the art board? The default value is `false`.
blurAmount		Number	The amount of blur to apply to the exported image. This value ranges from 0.0 to 2.0. The default value is 0.0.
horizontalScale		Number	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. The default value is 100.0.
matte		Boolean	Should the art board be matted with a color? The default value is `true`.
matteColor		RGBColor object	The color to use when matting the art board. The default value is `white`.
optimization		Boolean	Should the exported image be optimized for web viewing? The default value is `true`.
qualitySetting		Number	The quality of the exported image. This value ranges from 0 to 100. The default value is 30.
saveAsHTML		Boolean	Should the exported image be saved with an accompanying HTML file? The default value is `false`.
typename	R/O	String	Returns the name of the referenced object.
verticalScale		Number	The vertical scaling factor to apply to the exported image, where 100.0 is 100%. The default value is 100.0.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

`ExportOptionsJPEG` can only be supplied in conjunction with the `exportFile` method.

It is not necessary to specify values for all properties. Default values will be provided for any properties not specified.

Example

This example demonstrates how to export the current document as a JPEG with specific options.

```
// This script exports the current document as a JPEG with specific options

   var exportOptions = new ExportOptionsJPEG();
   var type = ExportType.JPEG;
   var fileSpec = new File("//temp/sample");
   var docRef = activeDocument;

   if (documents.length > 0)
   {
      exportOptions.antiAliasing = false;
      exportOptions.qualitySetting = 70;
      docRef.exportFile(fileSpec, type, exportOptions);
   }
```

ExportOptionsPhotoshop

Options which may be supplied when exporting a document as a Photoshop file. See the `exportFile` method for additional details.

Properties

Property	R/O	Value type	What it is
antiAliasing		Boolean	Should the exported image be anti-aliased? Default: `true`.
compoundShapes		Boolean	Export compound shapes as shape layers? Default: true
editableText		Boolean	Export text objects as editable text layers: Default: true
embedICCProfile		Boolean	Should an ICC profile be embedded in the exported file. Default: false
hiddenLayers		Boolean	Should hidden layers be exported? Default: false
imageColorSpace		ImageColorSpace constant	The color space of the exported file.
imageMap		Boolean	For RGB documents, should the image maps be preserved in ImageReady 3.0 format. Default: true
nestedLayers		Boolean	Should nested layers be exported. Default: true
resolution		Integer (72 - 2400)	The resolution of the exported file (in dots per inch). Default: 150
slices		Boolean	Should slice data be preserved? Default: true
typename	R/O	String	Returns the name of the referenced object.
warnings		Boolean	Should a warning dialog be display because of conflicts in the export settings. Default: true
writeLayers		Boolean	Should the document layers be presented in the exported document. Default: true

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

`ExportOptionsPhotoshop` can only be supplied in conjunction with the `exportFile` method.

It is not necessary to specify values for all properties. Default values will be provided for any properties not specified.

Example

This example exports the current document as a Photoshop file with specific options.

```
// This script exports the current document as a Photoshop file with layers.

   var exportOptions = new ExportOptionsPhotoshop();
   var type = ExportType.PHOTOSHOP;
   var fileSpec = new File("//temp/sample.psd");
   var docRef = activeDocument;

   if (documents.length > 0)
   {
      exportOptions.resolution = 150;
      docRef.exportFile(fileSpec, type, exportOptions);
   }
```

ExportOptionsPNG24

Options which may be supplied when exporting a document as an 8-bit PNG file. See the `exportFile` method for additional details.

Properties

Property	R/O	Value type	What it is
antiAliasing		Boolean	Should the exported image be anti-aliased? The default value is `true`.
artBoardClipping		Boolean	Should the exported image be clipped to the art board? The default value is `false`.
horizontalScale		Number	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. The default value is 100.0.
matte		Boolean	Should the art board be matted with a color? The default value is `true`.
matteColor		RGBColor object	The color to use when matting the art board. The default value is `white`.
saveAsHTML		Boolean	Should the exported image be saved with an accompanying HTML file? The default value is `false`.
transparency		Boolean	Should the exported image use transparency? The default value is `true`.
typename	R/O	String	Returns the name of the referenced object.
verticalScale		Number	The vertical scaling factor to apply to the exported image, where 100.0 is 100%. The default value is 100.0.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

`ExportOptionsPNG24` can only be supplied in conjunction with the `exportFile` method.

It is not necessary to specify values for all properties. Default values will be provided for any properties not specified.

Example

This example exports the current document as a PNG24 file.

```
// This script exports the current document as a PNG24 with specific options

   var docRef = activeDocument;
   var exportOptions = new ExportOptionsPNG24();
   var type = ExportType.PNG24;
   var fileSpec = new File("//temp/sample.png");

   if (documents.length > 0)
   {
      exportOptions.antiAliasing = false;
      exportOptions.transparency = false;

      docRef.exportFile(fileSpec, type, exportOptions);
   }
```

ExportOptionsPNG8

Options which may be supplied when exporting a document as an 8-bit PNG file. See the `exportFile` method for additional details.

Properties

Property	R/O	Value type	What it is
antiAliasing		Boolean	Should the exported image be anti-aliased? The default value is `true`.
artBoardClipping		Boolean	Should the exported image be clipped to the art board? The default value is `false`.
colorCount		Number	The number of colors in the exported image's color table. Acceptable values range from 2 to 256. The default value is 128.
colorDither		ColorDitherMethod constant	The method used to dither colors in the exported image. The default value is `aiDiffusionDither`.
colorReduction		ColorReductionMethod constant	The method used to reduce the number of colors in the exported image. The default value is `selective`.
ditherPercent		Number	How much should the colors of the exported image be dithered, where 100.0 is 100%.
horizontalScale		Number	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. The default value is 100.0.
interlaced		Boolean	Should the exported image be interlaced? The default value is `false`.
matte		Boolean	Should the art board be matted with a color? The default value is `true`.
matteColor		RGBColor object	The color to use when matting the art board. The default value is `white`.

Property	R/O	Value type	What it is
saveAsHTML		Boolean	Should the exported image be saved with an accompanying HTML file? The default value is `false`.
transparency		Boolean	Should the exported image use transparency? The default value is `true`.
typename	R/O	String	Returns the name of the referenced object.
verticalScale		Number	The vertical scaling factor to apply to the exported image, where 100.0 is 100%. The default value is 100.0.
webSnap		Number	How much should the color table be changed to match the web palette, where 100 is maximum. The default value is 0.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

`ExportOptionsPNG8` can only be supplied in conjunction with the `exportFile` method.

It is not necessary to specify values for all properties. Default values will be provided for any properties not specified.

Example

This example exports the current document as a PNG8 file.

```
// This script exports the current document as a PNG8 with specific options

   var docRef = activeDocument;
   var exportOptions = new ExportOptionsPNG8();
   var type = ExportType.PNG8;
   var fileSpec = new File("//temp/sample.png");

   if (documents.length > 0)
   {
      exportOptions.antiAliasing = false;
      exportOptions.interlaced = true;

      activeDocument.exportFile(fileSpec, type, exportOptions);
   }
```

ExportOptionsSVG

Options which may be supplied when exporting a document as a SVG file. See the `exportFile` method for additional details.

Properties

Property	R/O	Value type	What it is
compressed		Boolean	Should the exported file be compressed? The default value is `false`.
coordinatePrecision		Number	The decimal precision for element coordinate values. This value can range from 1 to 7. The default value is 3.
cssProperties		SVGCSSPropertyLocation constant	How should the CSS properties of the document be included in the exported file?
documentEncoding		SVGDocumentEncoding constant	How should the text in the document be encoded?
embedAllFonts		Boolean	Embed all fonts used by the document in the saved file?
embedRasterImages		Boolean	Embed raster images contained in the document in the saved file?
fontSubsetting		SVGFontSubsetting constant	What font glyphs should be included in the export file?
includeFileInfo		Boolean	Should file information be saved with exported file? Default: false
includeVariablesAndDatasets		Boolean	Should variables and datasets be saved with exported file? Default: false
optimizeForSVGViewer		Boolean	Should the exported file be optimized for the SVG Viewer? Default: false
preserveEditability		Boolean	Should Illustrator editing capabilities be preserved when exporting the document? Default: false
slices		Boolean	Should slice data be exported with the file? Default: false
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

`ExportOptionsSVG` can only be supplied in conjunction with the `exportFile` method.

It is not necessary to specify values for all properties. Default values will be provided for any properties not specified.

Example

This example exports the current document as a SVG file.

```
// This script exports the current document as a SVG with specific options

   var docRef = activeDocument;
   var exportOptions = new ExportOptionsSVG();
   var type = ExportType.SVG;
   var fileSpec = new File("//temp/sample.svg");

   if (documents.length > 0)
   {
      exportOptions.embedRasterImages = true;
      exportOptions.embedAllFonts = true;

      docRef.exportFile(fileSpec, type, exportOptions);
   }
```

File

The description of a file to be opened or saved with saveAs() method. File names are specified in URL format.

Properties

Property	R/O	Value type	What it is
alias	R/O	Boolean	Value is true if the object refers to a file system alias.
created	R/O	Date	The creation date of the object.
creator	R/O	String	The Macintosh file creator as a four-character string. For Windows, the return value is "????"
encoding		String	Gets or sets the encoding for subsequent read/write operations. (See notes below)
eof	R/O	Boolean	This value is true if the current read/write position is at the end of the file.
error		String	Contains a message describing the last file system error. Setting this value clears any error message and resets the error bit for opened files.
exists		Boolean	Value is true if the path name of this object refers to an existing file, folder, or file system alias.
fs	R/O	String	The name of the file system.
fsName	R/O	String	The file-system specific name of the object.
fullName	R/O	String	The full path name for the object in URL notation.
hidden		Boolean	Value is true if the object is invisible. Assigning a boolean value sets or clears this attribute.
linefeed		String	How line feed characters are written ("macintosh", "unix", or "windows"
length		Number	The size of the file. When setting the file size, the file must not be opened.

Property	R/O	Value type	What it is
modified	R/O	Date	The date when the object was last modified. If the object does not refer to a folder or file on disk, the value is null.
name		String	The name of the object without the path specification.
parent	R/O	File	The folder object containing this object. If this object already is the root folder of a volume, this value is null.
path		String	The path portion of the file name.
readonly		Boolean	If this value is true, the file cannot be altered or deleted. Folders may refuse to have this attribute set.
type	R/O	String	The type of the file. (See notes below)

Methods

Method	Parameter type	Returns	What it does
close()	none	Boolean	Closes the file. Returns true if the close operation is successful.
copy(target)	string or fileSpec object	Boolean	Copies the file to the specified location. Returns true if the file was successfully written. If there is a file at the target location, it is overwritten.
open(mode, type, creator	 string: r[ead] w[rite] e[dit] String (Macintosh only) String (Macintosh only)	Boolean	Opens the file for read, write, or modification. Returns true if the file is opened successfully.
read(bytes)	number	String	Reads the number of bytes specified. If no number is provided, reads the entire file.

Method	Parameter type	Returns	What it does
readch()	none	String	Read one single character. Line feeds are recognized as CR, LF, CRLF or LFCR pairs. Use this method to read files that contain multibyte character sequences
readln()	none	String	Reads one line of text. Line feeds are recognized as CR, LF, CRLF or LFCR pairs. Use this method to read files that contain multibyte character sequences
rename(newName)	string	Boolean	Changes the name of the file. Returns true if the rename operation was successful.
remove()	none	Boolean	Deletes the file or folder.
resolve()	none	File, Folder, or null	Resolves the file system alias for this object.
seek(pos, mode)	 number number	Boolean	Move to a specified position in the file. Position is determined by mode: 0 = seek to absolute position 1 = seek relative to current position 2 = seek backwards from end of file.
tell()	none	Number	Returns current position in the file.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.
write (textString 1 [,textString 2] | [,textString n])	string(s)	Boolean	All text strings are concatenated together to form the string to be written. Returns true if the operation is successful.

Method	Parameter type	Returns	What it does
writeln(textString 1 [,textString 2] | [,textString n])	string(s)	Boolean	Writes the specified string to the file and appends a Line Feed sequence. All text strings are concatenated together to form the string to be written. Returns true if the operation is successful.

Notes

The `remove()` method deletes the file or folder immediately. It does not move the referenced file or folder to the Recycle Bin. The `remove()` method cannot be undone. It is recommended that you use the confirm() method to prompt the user before completing this operation.

If the `resolve()` method is successful, it returns the specified file or folder object. If the specified object is not an alias or if the file or folder does not exist, null is returned.

The encoding property is one of several predefined constants that follow the common Internet encoding names. Valid names are: UCS-2, X-SJIS, ISO-8851-9, ASCII. A special encoder, BINARY, is used to read binary files. This encoder stores each byte of a file as one Unicode character regardless of an encoding. When writing, the lower byte of each Unicode character is treated as a single byte to write.

The Macintosh file type is a four-character string. On Macintosh, the specific file type is returned. For Windows, "appl" is returned for .exe files, "shlb" for .dll files, "alis" for aliases and "TEXT" for any other files.

Example

```
// This script checks to determine if the contents of the referenced
// folder are files or subfolders. It writes the results in a file.

   var theFolder = "//Pictures/Friends"
   var folderSpec = new Folder(theFolder);
   var folderArray = new Array();
   var fileArray = new Array();
   var folderCount = 0;
   var fileCount = 0;
   myPictures = folderSpec.getFiles()
   for (i = 0; i < myPictures.length; i++)
   {
      theFile = myPictures[i];
      if (theFile instanceof File)
      {
         fileArray[fileCount] = theFile;
         fileCount++
      }
         else
         {
         folderArray[folderCount] = theFile
         folderCount++
         }
   }

// Create a file that lists the folders and files.

   var reportFile = new File("//Documents/Filelist.txt");
   reportFile.open("write");
   reportFile.writeln("Folders:");
   for (i = 0; i < folderCount; i++)
   {
      reportFile.writeln("" + folderArray[i]);
   }

   reportFile.writeln("Files:");
   for (i = 0; i < fileCount; i++)
   {
      reportFile.writeln("" + fileArray[i]);
   }
   reportFile.close();

   alert("File report:\n" +
      "\nFolders: " + folderCount +
      "\nFiles: " + fileCount);
```

Folder

The description of a folder to be opened or saved with saveAs() method. Folder names are specified in URL format.

Properties

Property	R/O	Value type	What it is
alias	R/O	Boolean	Value is true if the object refers to a file system alias.
created	R/O	Date	The creation date of the object.
current		File object	Returns the current folder object. Assigning either a folder object or a string containing the new path name sets the current folder.
error		String	Contains a message describing the last file system error. Setting this value clears any error message and resets the error bit for opened files.
exists		Boolean	Value is true if the path name of this object refers to an existing file, folder, or file system alias.
fs	R/O	String	The name of the file system.
fsName	R/O	String	The file-system specific name of the object.
fullName	R/O	String	The full path name for the object in URL notation.
hidden		Boolean	Value is true if the object is invisible. Assigning a boolean value sets or clears this attribute.
modified	R/O	Date	The date when the object was last modified. If the object does not refer to a folder or file on disk, the value is null.
name		String	The name of the object without the path specification.
parent	R/O	File object	The folder object containing this object. If this object already is the root folder of a volume, this value is null.
path		String	The path portion of the file name.

Property	R/O	Value type	What it is
readonly		Boolean	If this value is true, the file cannot be altered or deleted. Folders may refuse to have this attribute set.
startup	R/O	File object	The folder containing the executable image of the running application.
system	R/O	File object	The folder containing the operating system files.
temp	R/O	File object	The default folder for storing temporary files.
trash	R/O	File object	The folder containing deleted items.

Methods

Method	Parameter type	Returns	What it does
create()	none	Boolean	Creates a folder at the specified path. Returns true if the operation is successful.
getFiles(mask)	String	Array	Gets a list of File and Folder objects contained in the referenced Folder object. (See notes below.)
rename(newName)	String	Boolean	Changes the name of the file. Returns true if the rename operation was successful.
remove()	none	Boolean	Deletes the file or folder.
resolve()	none	File, Folder, or null	Resolves the file system alias for this object.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

The `remove()` method deletes the file or folder immediately. It does not move the referenced file or folder to the Recycle Bin. The `remove()` method cannot be undone. It is recommended that you prompt the user for confirmation before completing this operation.

If the `resolve()` method is successful, it returns the specified file or folder object. If the specified object is not an alias or if the file or folder does not exist, null is returned.

The "mask" parameter of the getFiles(mask) method is the search mask for the file names. It may contain wildcards (question marks and asterisks) and is preset to * to find all files. Alternatively, you may supply a function. This function is called with a File or Folder object for every file or folder in the directory search. If the function returns true, the object is added to the array.

The return value is an array of File and Folder objects that correspond to the files found in the search. The return value is null if the file or folder does not exist.

Example

See example under File for a sample of using the Folder object.

Gradient

A gradient definition contained in a document.

Properties

Property	R/O	Value type	What it is
gradientStops	R/O	GradientStops collection object	The gradient stops contained in this gradient.
name		String	The gradient's name.
parent	R/O	Document object	The document that contains this gradient.
type		GradientType constant	The kind of the gradient, either radial or linear.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
remove()	none	Nothing	Removes the referenced object from the document.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Illustrator's `Gradient` object represents a gradient as defined in the Illustrator document. Additional gradients may be created by the user within Illustrator or via a script.

Example

This example shows how you can create a new gradient and apply it as a fill pattern to the frontmost path item.

```
// This example shows how you can create a new gradient and apply it to the
// frontmost path item in the document


if (documents.length > 0)
{
    // Create a color for both ends of the gradient

    startColorRGB = new RGBColor();
    startColor = new Color();
    endColorRGB = new RGBColor();
    endColor = new Color();

    startColorRGB.red = 0;
    startColorRGB.green = 100;
    startColorRGB.blue = 255;
    startColor.rgb = startColorRGB;

    endColorRGB.red = 220;
    endColorRGB.green = 0;
    endColorRGB.blue = 100;
    endColor.rgb = endColorRGB;

    // Create a new gradient
    // A new gradient always has 2 stops

    newGradient = activeDocument.gradients.add();
    //newGradient.name = "Gradient created from script";
    newGradient.type = GradientType.LINEAR;

    // Modify the first gradient stop.

    locationSpecification = newGradient.gradientStops[0];
    locationSpecification.rampPoint = 30;
    locationSpecification.midPoint = 60;
    locationSpecification.color = startColor;

    // Modify the last gradient stop.
    // The MidPoint for the last gradient stop is ignored

    locationSpecification = newGradient.gradientStops[1];
    locationSpecification.rampPoint = 80;
    locationSpecification.color = endColor;

    // construct an Illustrator.Color object referring to the
    // newly created gradient

    colorOfGradient = new GradientColor();
    pathFillColor = new Color();
    colorOfGradient.gradient = newGradient;
    pathFillColor.gradient = colorOfGradient;

    // now get the frontmost path item and apply the new gradient
    // as its fill

    topPath = activeDocument.pathItems[0];
    topPath.filled = true;
    topPath.fillColor = pathFillColor;
}
```

GradientColor

A gradient color specification, used in conjunction with the `Gradient` property of the `Color` specification.

Properties

Property	R/O	Value type	What it is
angle		Number	The gradient vector angle (in degrees).
gradient		Gradient object	Reference to the object defining the gradient.
hiliteAngle		Number	The gradient hilite vector angle (in degrees).
hiliteLength		Number	The gradient hilite vector length.
length		Number	The gradient vector length.
matrix		Matrix object	An additional transformation matrix to manipulate the gradient path.
origin		Array (of 2 numbers)	The gradient vector origin.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

A `GradientColor` can be created using a reference to an existing gradient in the document. If no existing gradient object is referenced, a default gradient will be supplied. An origin is used to specify the center point of the gradient in this specific gradient color. Numeric values are used to specify the gradient vector angles and lengths. A matrix may be specified to further transform the gradient color.

Example

The following script obtains the gradient called “Black, White Radial” from the current document and changes the color of the first gradient stop. The Gradient “Black, White Radial” is one of the default gradients that appear when you create a new Illustrator document.

```
// This example shows how to change the color for the first gradient stop
// of the gradient called "Black, White Radial" in the active document


if (documents.length > 0)
{
   // Get a reference to the gradient that you want to delete

    firstGradient = activeDocument.gradients["Black, White Radial"];

   // Create the new color

    startRGBColor = new RGBColor();
    startColor = new Color();

    startRGBColor.red = 255;
    startRGBColor.green = 238;
    startRGBColor.blue = 98;

    startColor.rgb = startRGBColor;

    firstGradient.gradientStops[0].color = startColor;
}
```

Gradients

The collection of gradients in a document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	gradient object	Creates a new object.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Illustrator's `Gradient` object represents a gradient as defined in the Illustrator document. Additional gradients may be created by the user within Illustrator or via a script.

Example

This example illustrates how you can remove a gradient from a document.

```
// This example shows how to delete the first gradient in the
// active document
if (documents.length > 0)
{
    activeDocument.gradients[0].remove();
}
```

GradientStop

A gradient stop definition contained in a specific gradient.

Properties

Property	R/O	Value type	What it is
color		Color object	The color linked to this gradient stop.
midpoint		Number	The midpoint key value is specified as a percentage from 0.0 to 100.0.
parent		Gradient object	The gradient that contains this gradient stop.
rampPoint		Number	The location of the color in the blend in a range from 0.0 to 100.0, where 100.0 is 100%.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
remove()	none	Nothing	Removes the referenced item from the document.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

The `GradientStop` object represents a point on a specific gradient defined in the document. Each gradient stop specifies a color change in the containing gradient.

See example under GradientColor for an example of how to use gradientStop.

GradientStops

A collection of gradient stops in a specific gradient.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
add()	none	GradientStop object	Creates a new object.
removeAll		Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Illustrator’s `GradientStop` object represents a point on a specific gradient defined in the Illustrator document. Each gradient stop specifies a color change in the containing gradient.

Example

This example illustrates how to add a new gradient stop to an existing gradient.

```
// This example shows how to add a gradient stop to a gradient

if (documents.length > 0 && activeDocument.gradients.length > 0)
{
   // Get a reference to the gradient that you want to change
    gradientToChange = activeDocument.gradients[0];

   // Get a reference to the gradient stop that is the last one
   // before you add a new gradient stop

    originalCount = gradientToChange.gradientStops.length;
    lastGradientStop = gradientToChange.gradientStops[originalCount-1];

   // add the new gradient stop
    newGradientStop = gradientToChange.gradientStops.add();

   // Set the values of the new gradient stop. We move the original last
   // gradient stop a bit to the left and
   // insert the new gradient stop at the old gradient stops position

    newGradientStop.rampPoint = lastGradientStop.rampPoint;
    lastGradientStop.rampPoint = lastGradientStop.rampPoint - 10;

   // Create a new color to apply to the newly created gradient stop
   // we choose a Gray tint value of 70%

    colorOfGradientStop = new GrayColor();
    newStopColor = new Color();
    colorOfGradientStop.gray = 70.0;
    newStopColor.gray = colorOfGradientStop;
    newGradientStop.color = newStopColor;
}
```

GraphItem

Any `graph` artwork object.

Properties

Property	R/O	Value type	What it is
artworkKnockout		KnockoutState constant	Is this object used to create a knockout? If so, what kind of knockout? You cannot set this value to aiKnockoutUnknown.
blendingMode		BlendModes constant	The mode used when compositing an object.
contentVariable		Variant	The content variable bound to the graphItem.
controlBounds	R/O	Array (of 4 numbers)	The bounds of the object including stroke width and controls.
editable	R/O	Boolean	Is this graphItem editable?
geometricBounds	R/O	Array (of 4 numbers)	The bounds of the object excluding stroke width.
height		Number	The height of the graph item.
hidden		Boolean	Is this graph item hidden?
isIsolated		Boolean	Is this object isolated?
layer	R/O	layer object	The layer to which this graph item belongs.
left		Number	The position of the left side of the graphItem.
locked		Boolean	Is this graph item locked?
name		String	The name of this graph item.
opacity		Numbers	The opacity of the object . The value is between 0.0 and 100.0.
parent	R/O	Layer object or GroupItem object	The parent of this object.
position		Array (of 2 numbers)	The position of the top left corner of the graph item.
selected		Boolean	Is this object selected?
sliced		Boolean	Is the graphItem sliced? Default: false
tags	R/O	Tags collection object	The tags contained in this graphItem.

Property	R/O	Value type	What it is
top		Number	The position of the top of the graphItem.
typename	R/O	String	The type of the graphItem.
url		String	The value of the Adobe URL tag assigned to this graph item.
visibilityVariable		Variable object	The visibility variable bound to the graphItem.
visibleBounds	R/O	Array (of 4 numbers)	The visible bounds of the graph item including stroke width.
width		Number	The width of the graph item.
zOrderPosition	R/O	Number	The position of this art object within the stacking order of the group or layer (parent) that contains the art object.

Methods

Method	Parameter type	Returns	What it does
duplicate()	none	GraphItem	Create a duplicate of the selected graphItem.
moveAfter(destination)	object	Nothing	Moves the graphItem behind the specified object.
moveBefore(destination)	object	Nothing	Moves the graphItem in front of the specified object.
moveToBeginning (destination)	object (document, layer, or groupItem)	Nothing	Moves the graphItem to the front of the specified container.
moveToEnd(destination)	object (document, layer, or groupItem)	Nothing	Moves the graphItem to the end of the specified container.
remove()	none	Nothing	Removes the graphItem from the document.
resize(scaleX, scaleY [,changePositions], [,changeFillPatterns], [,changeFillGradients], [,changeStrokePattern], [,changeLineWidths] [,scaleAbout])	number boolean boolean boolean boolean boolean Transformation constant	Nothing	Scales the art object where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%.

Method	Parameter type	Returns	What it does
rotate(angle [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,rotateAbout])	 number boolean boolean boolean boolean Transformation constant	Nothing	Rotates the art object relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
toString()	none	String	Returns a character representation of the referenced object.
transform transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])	 Matrix object boolean boolean boolean boolean number Transformation constant	Nothing	Transforms the art object by applying a transformation matrix.
translate([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])	 number number boolean boolean boolean boolean	Nothing	Repositions the art object relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
zOrder(ZOrderMethod)	ZOrderMethod constant	Nothing	Arranges the art object's position in the stacking order of the group or layer (Parent) of this object.

Notes

It is not necessary to set the type of the contentVariable before binding. Illustrator automatically set the type to `GRAPH`.

It is not necessary to set the type of the visibilityVariable before binding. Illustrator automatically set the type to `VISIBILITY`.

GraphItems

A collection of graph items. The `GraphItems` object gives you access to all the graph art objects in an Illustrator document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
removeAll()	none	Nothing	Deletes all objects in the collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

GrayColor

A gray color specification, used in conjunction with the `Gray` property of the `Color` specification.

Properties

Property	R/O	Value type	What it is
gray		Number	The tint of the gray as a value in the range 0.0 - 100.0, where 0.0 is black and 100.0 is white.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This example illustrates how to change the color of the first word in the active document to a shade of gray.

```
// the following script shows how to set the color of the first
// word in the active document to a shade of gray

// Get a reference to the first word in the active document

if (documents.length > 0 && activeDocument.textArtItems.length > 0)
{
    text = activeDocument.textArtItems[0].textRange();
    firstWord = text.words[0];

   // Create the new color

    grayColorOfWord = new GrayColor();
    textColor = new Color;
    grayColorOfWord.gray = 45;
    textColor.gray = grayColorOfWord;

    firstWord.filled = true;
    firstWord.fillColor = textColor;
}
```

GroupItem

A grouped set of art objects.

Properties

Property	R/O	Value type	What it is
artworkKnockout		KnockoutState constant	Is this object used to create a knockout? If so, what kind of knockout?
blendingMode		BlendModes constant	The mode used when compositing an object.
clipped		Boolean	Is the group clipped to its first path item?
compoundPathItems	R/O	CompoundPathItems collection object	The compound path items contained in this group.
controlBounds	R/O	Array (of 4 numbers)	The bounds of the object including stroke width and controls.
editable	R/O	Boolean	Is this item editable?
geometricBounds	R/O	Array (of 4 numbers)	The bounds of the object excluding stroke width.
graphItems	R/O	GraphItems collection object	The graph items contained in this document.
groupItems	R/O	GroupItems collection object	The group items contained in this document.
height		Number	The height of the group item.
hidden		Boolean	Is this group item hidden?
isIsolated		Boolean	Is this object isolated?
layer	R/O	Layer object	The layer to which this group item belongs.
left		Number	The position of the left side of the item.
locked		Boolean	Is this group item locked?
meshItems	R/O	MeshItems collection object	The mesh items contained in this group.
name		String	The name of this group item.
opacity		Number	The opacity of the object . The value is between 0.0 and 100.0.
parent	R/O	Layer object or GroupItem object	The parent of this object.

Property	R/O	Value type	What it is
pathItems	R/O	PathItems collection object	The path items contained in this group.
placedItems	R/O	PlacedItems collection object	The placed items contained in this group.
pluginItems	R/O	PluginItems collection object	The plugin items contained in this group.
position		Array (of 2 numbers)	The position of the top left corner of the group item.
rasterItems	R/O	RasterItems collection object	The raster items contained in this group.
selected		Boolean	Is this group item selected?
sliced		Boolean	Is the item sliced? Default: false
symbolItems		SymbolItems collection object	The SymbolItem objects in this document.
tags	R/O	Tags collection object	The tags contained in this group.
textArtItems	R/O	TextArtItems collection object	The text art items contained in this group.
top		Number	The position of the top of the item.
typename	R/O	String	Returns the name of the referenced object.
url		String	The value of the Adobe URL tag assigned to this group item.
visibilityVariable		Variable object	The visibility variable bound to the item.
visibleBounds	R/O	Array (of 4 numbers)	The visible bounds of the group item including stroke width.
width		Number	The group of the page item.
zOrderPosition	R/O	Number	The position of this art object within the stacking order of the group or layer (Parent) that contains the art object.

Methods

Method	Parameter type	Returns	What it does
duplicate()	none	GroupItem object	Creates a duplicate of the selected item.
moveAfter(destination)	object	Nothing	Moves the item behind the specified object.

Method	Parameter type	Returns	What it does
moveBefore(destination)	object	Nothing	Moves the item in front of the specified object.
moveToBeginning (destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the front of the specified container.
moveToEnd(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the end of the specified container.
remove()	none	Nothing	Removes the referenced item from the document.
resize(scaleX, scaleY [,changePositions], [,changeFillPatterns], [,changeFillGradients], [,changeStrokePattern], [,changeLineWidths] [,scaleAbout])	number boolean boolean boolean boolean boolean Transformation constant	Nothing	Scales the art object where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%.
rotate(Angle [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,rotateAbout])	number boolean boolean boolean boolean Transformation constant	Nothing	Rotates the art object relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.
transform transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])	Matrix object boolean boolean boolean boolean number Transformation constant	Nothing	Transforms the art object by applying a transformation matrix.

Method	Parameter type	Returns	What it does
translate([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])	 number number boolean boolean boolean boolean	Nothing	Repositions the art object relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
zOrder(ZOrderMethod)	ZOrderMethod constant	Nothing	Arranges the art object's position in the stacking order of the group or layer (Parent) of this object.

Notes

Group items can contain all of the same page items that a layer can contain, including other nested groups.

Paths contained within a group or compound path in a document will be returned as individual paths when a script asks for the paths contained in the document. However, paths contained in a group or compound path will not be returned when a script asks for the paths in a layer which contains the group or compound path.

Example

It is easy to modify all of the objects contained in a group. This example demonstrates how to simplify your operations on multiple objects by creating group to contain them.

```
// The following script show how to create new art in a separate group
// Create a new group in the active document. This will be the group
// that holds the new triangle art

if (documents.length > 0)
{
   triangleGroup = activeDocument.groupItems.add();

   // Create a triangle and add text. All new art are created inside a group

    trianglePath = triangleGroup.pathItems.add();
    trianglePath.setEntirePath(Array(Array(100, 100), Array(300, 100),
        Array(200, Math.tan(1.0471975) * 100 + 100)));
   trianglePath.stroked = true;
   trianglePath.strokeWidth = 3;

    captionText = triangleGroup.textArtItems.add();
    captionText.position = Array(100, 100);
    captionText.contents = "A triangle";
}
```

GroupItems

The collection of grouped art objects in the document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	GroupItem object	Creates a new object.
createFromFile(imageFile)	File object	GroupItem object	Places an external vector art file as a group item in the document.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

The following script shows how you can import a PDF document using the CreateFromFile function. Before running this script you have to create a one page PDF file and put it in the following location: "/temp/testfile1.pdf"

```
// This example shows how to create a group from a file
// In order to run this example you need a PDF file at
// the path "//temp/testfile1.pdf"

if (documents.length > 0)
{
   theFileSpec = new File("//temp/testfile1.pdf");
   importedGroup = activeDocument.groupItems.createFromFile(theFileSpec);
}
```

IllustratorSaveOptions

Options which may be supplied when saving a document as an Illustrator file. See the `Save` method for additional details.

Properties

Property	R/O	Value type	What it is
compatibility		Compatibility constant	Specifies the version of Illustrator file format to create.
compressed		Boolean	Should the saved file be compressed. Default: true (Illustrator version 10 or later)
embedAllFonts		Boolean	Embed all fonts used in the document in the saved file? (Illustrator version 9 or later)
embedICCProfile		Boolean	Embed document’s ICC profile in the saved file? (Illustrator version 9 or later)
embedLinkedFiles		Boolean	Embed linked image files in the saved file? (Illustrator version 7 or later)
flattenOutput		OutputFlattening constant	How should transparency be flattened for older file format versions. (Versions before Illustrator 9)
fontSubsetThreshold		Number	Include a subset of fonts when less than this percentage of characters is used in the document. (Illustrator version 9 or later)
japaneseFileFormat		Boolean	Save using Japanese version of file format. (Illustrator versions 3 - 5 only)
pdfCompatibility		Boolean	Save as PDF compatible file? (Illustrator version 10 or later)
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What does it do
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Illustrator save options can only be supplied in conjunction with the `saveAs` method.

It is not necessary to specify values for all properties. Default values will be provided for any properties not specified.

Example

This script illustrates how to save the frontmost document as an Illustrator file.

```
// This script saves the active document as Illustrator 7 format
// Opacity is flattened with the preserve appearance option

if (documents.length > 0)
{
    saveOptions = new IllustratorSaveOptions;
    fileSpec = new File("//temp/Ai7Sample.ai");

    saveOptions.compatibility = Compatibility.ILLUSTRATOR7;
    saveOptions.flattenOutput = OutputFlattening.PRESERVEAPPEARANCE;
    activeDocument.saveAs(fileSpec, saveOptions);
}
```

Layer

A layer in an Illustrator document. Layers may contain nested layers, which are called sublayers in the user interface.

Properties

Property	R/O	Value type	What it is
artworkKnockout		KnockoutState constant	Is this object used to create a knockout? If so, what kind of knockout? You cannot set this value to UNKNOWN.
blendingMode		BlendModes constant	The mode used when compositing an object.
color		RGBColor object	The layer's selection mark color.
compoundPathItems	R/O	CompoundPathItems collection object	The compound path items contained in this layer.
dimPlacedImages		Boolean	Are placed images to be rendered as dimmed in this layer?
graphItems	R/O	GraphItems collection object	The Graph items contained in this layer.
groupItems	R/O	GroupItems collection object	The group items contained in this layer.
hasSelectedArtwork		Boolean	Is any object in this layer selected? Setting this property to false deselects all objects in the layer.
isIsolated		Boolean	Is this object isolated?
layers	R/O	Layers collection object	The layers contained in this layer.
locked		Boolean	Is this layer editable? Setting this property to `true` locks the layer.
meshItems	R/O	MeshItems collection object	The mesh items contained in this layer.
name		String	The name of this layer.
opacity		Number	The opacity of the layer. The value is between 0.0 and 100.0.
pageItems	R/O	PageItems collection object	The page items contained in this layer.
parent	R/O	Document object or layer Object	The document or layer that contains this layer.
pathItems	R/O	PathItems collection object	The path items contained in this layer.

Property	R/O	Value type	What it is
placedItems	R/O	PlacedItems collection object	The placed items contained in this layer.
pluginItems	R/O	PluginItems collection object	The plugin items contained in this layer.
preview		Boolean	Is this layer displayed using preview mode?
printable		Boolean	Is this layer printed when printing the document?
rasterItems	R/O	RasterItems collection object	The raster items contained in this layer.
sliced		Boolean	Is the item sliced? Default: false
symbolItems		SymbolItems collection object	The SymbolItems contained in the layer.
textArtItems	R/O	TextArtItems collection object	The text art items contained in this layer.
typename	R/O	String	Returns the name of the referenced object.
visible		Boolean	Is this layer visible?
zOrderPosition	R/O	Number	The position of this layer within the stacking order of layers in the document.

Methods

moveAfter(destination)	object	Nothing	Moves the item behind the specified object.
moveBefore(destination)	object	Nothing	Moves the item in front of the specified object.
moveToBeginning(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the front of the specified container.
moveToEnd(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the end of the specified container.
remove()	none	Nothing	Removes the referenced item from the document.

toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.
zOrder(ZOrderMethod)	ZOrderMethod constant	Nothing	Arranges the art object's position in the stacking order of the group or layer (parent) of this object.

Notes

Illustrator's `layer` object contains all of the page items in the specific layer as elements. Your script can access page items as elements of either the `Layer` object or as elements of the `document` object. When accessing page items as elements of a layer, only objects in that layer can be accessed. To access page items throughout the entire document, be sure to refer to them as contained by the document.

Example

This example illustrates how to move the bottom layer to the front.

```
// This example shows how to move the bottom layer to the top

// Get a reference to the layers, and obtain the total number

if (documents.length > 0)
{
    countOfLayers = activeDocument.layers.length;

     if (countOfLayers > 1)
    {
        // Move the bottom layer to the front
        bottomLayer = activeDocument.layers[countOfLayers-1];
        bottomLayer.zOrder(ZOrderMethod.BRINGTOFRONT);
    }
    else
    {
          alert("The active document only has only 1 layer")
     }

}
```

Layers

The collection of layers in the document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
add()	none	Layer object	Creates a new layer in the document.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Illustrator's `Layer` object contains all of the page items in the specific layer. Your script can access objects through the `Layer` object or through the `document` object.

Example

This example illustrates how to delete all layers whose name starts with the word “Temporary” in all open documents.

```
// Example of how to delete all layers
// whose name begins with "Temporary" in all open documents

// loop through all open documents
for (i=0; i < documents.length; i++)
{
   targetDocument = documents[i];
   countOfLayers = targetDocument.layers.length;

   // Loop through layers from the back because this way we don’t change
   // the index of layers that have not been checked when we remove a layer

    for (j=countOfLayers-1; j>=0; j--)
   {
        targetLayer = targetDocument.layers[j];
        layerName = targetLayer.name;
        if (layerName.substring(0, 9) == "Temporary")
       {
            targetDocument.layers[j].remove();
        }
    }
}

alert("Layer count is: " + activeDocument.layers.length);
```

Matrix

A transformation matrix specification, used to transform the geometry of objects.

Properties

Property	R/O	Value type	What it is
mValueA		Number	Matrix property a.
mValueB		Number	Matrix property b.
mValueC		Number	Matrix property c.
mValueD		Number	Matrix property d.
mValueTX		Number	Matrix property tx.
mValueTY		Number	Matrix property ty.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

This class is used to define a record which contains the component values of an Illustrator transformation matrix. It is used for specifying and retrieving matrix information from an Illustrator document or from page items in a document.

Matrices are used in conjunction with the `transform` method and as a property of a number of objects. A matrix specifies how to transform the geometry of an object. You can generate an original matrix using the Application object methods `getTranslationMatrix()`, `getScaleMatrix()`, or `getRotationMatrix()`.

A `Matrix` is a record containing the matrix values, not a reference to a matrix object. The matrix commands listed above operate on the values of a matrix record. If a command modifies a matrix, a modified matrix record is returned as the result of the command. The original matrix record passed to the command is not modified.

Example

If you need to apply multiple transformations to objects it is more efficient to use the matrix suite than to apply the transformations one at a time. The following script demonstrates how to combine multiple matrices together.

```
// This example shows how to apply 2 tranformations to all art in a
// document using the matrix command. This is more efficient than performing
// these transformations one at a time.

// move art half an inch to the right and 1.5 inch up on the page

if (documents.length > 0)
{
   moveMatrix = getTranslationMatrix(0.5, 1.5);

   // Add a rotation to the translation. We rotate 10 degrees
   // counter clockwise
   totalMatrix = concatenateRotationMatrix(moveMatrix, 10);

   // apply the transformation to all art in the document

   for (i=0; i < activeDocument.pageItems.length; i++)
   {
      theArtItem = activeDocument.pageItems[i];
      theArtItem.transform(totalMatrix);
   }
}
```

MeshItem

A gradient mesh art object.

Properties

Property	R/O	Value type	What it is
artworkKnockout		KnockoutState constant	Is this object used to create a knockout? If so, what kind of knockout?
blendingMode		BlendModes constant	The mode used when compositing an object.
controlBounds	R/O	Array (of 4 numbers)	The bounds of the object including stroke width and controls.
editable	R/O	Boolean	Is this item editable?
geometricBounds	R/O	Array (of 4 numbers)	The bounds of the object excluding stroke width.
height		Number	The height of the mesh item.
hidden		Boolean	Is this mesh item hidden?
isIsolated		Boolean	Is this object isolated?
layer	R/O	Layer object	The layer to which this mesh item belongs.
left		Number	The position of the left side of the item.
locked		Boolean	Is this mesh item locked?
name		String	The name of this mesh item.
opacity		Number	The opacity of the object . The value is between 0.0 and 100.0.
parent	R/O	Layer object or GroupItem object	The parent of this object.
position		Array (of 2 numbers)	The position of the top left corner of the mesh item.
selected		Boolean	Is this mesh item selected?
sliced		Boolean	Is the item sliced? Default: false
tags	R/O	Tags collection object	The tags contained in this mesh item.
top		Number	The position of the top of the item.
typename	R/O	String	Returns the name of the referenced object.

Property	R/O	Value type	What it is
url		String	The value of the Adobe URL tag assigned to this mesh item.
visibilityVariable		Variable object	The visibility variable bound to the item.
visibleBounds	R/O	Array (of 4 numbers)	The visible bounds of the mesh item including stroke width.
width		Number	The width of the mesh item.
zOrderPosition	R/O	Number	The position of this art object within the stacking order of the group or layer (Parent) that contains the art object.

Methods

Method	Parameter type	Returns	What it does
duplicate()	none	Item	Creates a duplicate of the selected item.
moveAfter(destination)	object	Nothing	Moves the item behind the specified object.
moveBefore(destination)	object	Nothing	Moves the item in front of the specified object.
moveToBeginning(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the front of the specified container.
moveToEnd(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the end of the specified container.
remove()	none	Nothing	Removes the referenced item from the document.
resize(scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])	number number boolean boolean boolean boolean boolean Transformation constant	Nothing	Scales the art object where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%.

Method	Parameter type	Returns	What it does
rotate(angle [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,rotateAbout])	 number boolean boolean boolean boolean Transformation constant	Nothing	Rotates the art object relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.
transform transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])	 matrix object boolean boolean boolean boolean number Transformation constant	Nothing	Transforms the art object by applying a transformation matrix.
translate([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])	 number number boolean boolean boolean boolean	Nothing	Repositions the art object relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
zOrder(ZOrderMethod)	ZOrderMethod constant	Nothing	Arranges the art object's position in the stacking order of the group or layer (parent) of this object.

Notes

You cannot create mesh items from a script. You may copy mesh items by using the duplicate() method and then using the one of the move methods (moveAfter(), moveBefore(), moveToBeginning(), and moveToEnd()), to place the item at the proper location.

Example

This script illustrates how to lock all Mesh items in the active document.

```
// Example of how to lock all meshItems in the frontMost document

if (documents.length > 0)
{
    for (i=0; i < activeDocument.meshItems.length; i++)
   {
        activeDocument.meshItems[i].locked = true;
    }
}
```

MeshItems

A collection of gradient mesh art objects.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Mesh items cannot be created from a script. You may copy mesh items by using the duplicate() method and then using the one of the move methods (moveAfter(), moveBefore(), moveToBeginning(), and moveToEnd()), to place the item at the proper location.

Example

The following script illustrates how to copy mesh items from one document to another. To run this script you need to have two open documents. One document should contain at least one mesh item, the other document can be empty. Make the empty document the frontmost before running the script.

```
// This script shows how to copy all MeshItems from one document to
// another document

//$.bp();

if (documents.length > 0)
{
   sourceDocument = documents[0];

   locationOffset = 0

   targetDocument = documents.add();

   for (i=0; i < sourceDocument.meshItems.length; i++)
   {
      newMeshItem = sourceDocument.meshItems[i].duplicate();
      newMeshItem.moveToBeginning(targetDocument);

      // Get a reference to the item that was just copied
      // into the document

      newMeshItem.position = Array(100, 40 + locationOffset);
      locationOffset = locationOffset + 50;
   }
}
```

PageItems

A collection of page items.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

The `PageItems` class gives you complete access to all the art objects in an Illustrator document in the following classes:

- CompoundPathItems
- GraphItems
- GroupItems
- MeshItems
- PathItems
- PlacedItems
- TextPath_PathItems
- RasterItems
- SymbolItems
- TextArtItems

PageItems may be referenced by Document, Layer, or Group.

When you access each of the individual items in one of these collections, the reference returns the particular class item with all of its properties. For example, if you use PageItem to reference a TextArtItem, the typename property will be TextArtItem and the toString() method will return [TextArtItem *itemname*].

Example

This example illustrates how to obtain all references to external files in the current document. The result is presented in a new Illustrator document. Before running this script, you must open a document that contains one or more linked images.

```
// The following script shows how to get all file-references
// using the PageItems object

if (documents.length > 0)
{
   var fileReferences = new Array(9);

    index = 0;
    sourceDocument = activeDocument;
    sourceName =sourceDocument.name;
    for (i=0; i<sourceDocument.pageItems.length; i++)
   {
        artItem = sourceDocument.pageItems[i];
        switch (artItem.typename)
      {
            case "PlacedItem":
                placedArt = artItem;
                fileReferences[index] = placedArt.file;
                index = index + 1;
                break;
            case "RasterItem":
                rasterArt = artItem;
                fileReferences[index] = rasterArt.file.fullName;
                index = index + 1;
                break;
        }
        if (index > 9)
      {
            alert("There are more than 10 references in this document.");
            return
        }
    }

   // Write the file references to a new document
    reportDocument = documents.add();

    fileNameText = reportDocument.textArtItems.add();
    fileNameText.position = Array(50, 520);
    fileNameText.contents = "File references in " + sourceName + ":";
    for (counter = 0; counter<index; counter++)
   {
        fileNameText = reportDocument.textArtItems.add();
        fileNameText.position = Array(65, 500 - 20 * counter);
        fileNameText.contents = fileReferences[counter];
    }
}
```

Paragraph

A single paragraph of text in the contents of a text art object.

Properties

Property	R/O	Value type	What it is
autoKerning		Boolean	Should a font's built-in kerning information be used?
baselineShift		Number	Baseline offset of text.
characters	R/O	Characters collection object	The characters contained in this text range.
clipping	R/O	Boolean	Is there a clipping path associated with the text art item containing this paragraph?
contents (default value)		String	The text contained in the text range.
defaultTabSize		Number	The default distance for tab stops.
desiredLetterSpacing		Number	The desired letter spacing. 100.0 is normal letter spacing.
desiredWordSpacing		Number	The desired word spacing. 100.0 is normal word spacing.
direction		CharacterDirection constant	The orientation of the characters in a vertical text block.
evenodd		Boolean	Should the even-odd rule be used to determine insideness?
fillColor		Color	Fill color of text
filled		Boolean	Should the text be filled?
fillOverprint		Boolean	Should the art beneath the text be overprinted?
firstLineIndent		Number	The indent of the first line.
font		String	The text face of the text.
hangingPunctuation		Boolean	Should punctuation appear outside the margins of the paragraph?
hyphenation		Boolean	Is hyphenation enabled for the paragraph?
justification		Justification constant	The paragraph alignment or justification.
leading		Number	The vertical leading of the text.
leftIndent		Number	The left indent of the paragraph's margin.

Property	R/O	Value type	What it is
length	R/O	Number	The number of character in the text.
limitConsecutiveHyphenations		Boolean	Is there a limit on the number of consecutive hyphenated lines in this paragraph?
maximumConsecutiveHyphens		Number	The maximum number of consecutive hyphenated lines.
maximumLetterSpacing		Number	The maximum letter. 100.0 is normal letter spacing.
maximumWordSpacing		Number	The maximum letter spacing. 100.0 is normal word spacing
minimumAfterHyphen		Number	The minimum number of characters after a hyphen.
minimumBeforeHyphen		Number	The minimum number of characters before a hyphen.
minimumLetterSpacing		Number	The minimum letter spacing.100.0 is normal letter spacing
minimumWordSpacing		Number	The minimum letter spacing.100.0 is normal word spacing
note	R/O	String	The note associated with this text.
offset	R/O	Number	Offset of selected text in text range (in characters).
orientation	R/O	TextOrientation constant	The orientation of the text. Use the TextPath class to alter this property.
parent	R/O	TextArtItem object	The parent of this object.
repeatedCharacterProcessing		Boolean	Should Repeated Character Processing be used?
resolution	R/O	Number	The resolution of the object (in dots per inch).
rightIndent		Number	The right indent of the paragraph’s margin.
scaling		Array (of 2 numbers)	The character scaling supplied as a point with the first coordinate as horizontal scale and the second coordinate as vertical scale, where 100.0 is 100%.
size		Number	Font size of text.
spaceBefore		Number	The spacing before this paragraph.
strokeCap		StrokeCap constant	The type of line capping.
strokeColor		Color object	The stroke color for the path.

Property	R/O	Value type	What it is
stroked		Boolean	Should the path be stroked?
strokeDashes		Array	Dash lengths. Set to an empty array for a solid line.
strokeDashOffset		Number	The default distance into the dash pattern at which the pattern should be started.
strokeJoin		StrokeJoin constant	Type of joints for the path.
strokeMiterLimit		Number	Are joins mitered (pointed) or beveled (squared-off)?
strokeOverprint		Boolean	Will art beneath a stroked object be overprinted?
strokeWidth		Number	Width of stroke.
textLines	R/O	TextLines collection object	The lines of text contained in this paragraph.
textPath	R/O	TextPath object	A reference to the text path associated with the text art item containing this text.
tracking		Number	The spacing between multiple characters.
typename	R/O	String	Returns the name of the referenced object.
words	R/O	Words collection object	The words contained in this paragraph.

Methods

Method	Parameter type	Returns	What it does
remove()	none	Nothing	Removes the referenced item from the document.
textRange([rangeStart] [,rangeEnd])	 number number	TextRange object	Returns a text range object referencing a substring of the current text range, where rangeStart is the beginning character position and rangeEnd is the ending position. The first character position is zero. If omitted, rangeStart defaults to 0. If omitted, rangeEnd defaults to the last character of the range.

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Illustrator's text can be accessed using the `Character`, `Word`, `TextLine`, `Paragraph` and `TextRange` classes. All text is contained within text art items.

The `Paragraph` class has additional properties that other related classes do not share, including properties for margins, hyphenation, and word/letter spacing.

Example

This script illustrates how to turn on hyphenation on for all paragraphs in the frontmost document.

```
// Example of how to set hyphenation to true for all paragraphs in
// the frontmost document

if (documents.length > 0)
{
    frontDocument = activeDocument;

    for (i=0; i<frontDocument.textArtItems.length; i++)
   {
        textArtTextRange = activeDocument.textArtItems[i].textRange();
        for (j=0; j<textArtTextRange.paragraphs.length; j++)
       {
            currentParagraph = textArtTextRange.paragraphs[j];
            currentParagraph.hyphenation = true;
        }
    }
}
```

Paragraphs

A collection of paragraphs.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	Paragraph object	Add a paragraph to the contents of a text art object.
addBefore()	none	Paragraph object	Adds a paragraph before the current paragraph selection or insertion point.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This script displays the total number of paragraphs contained in all the `textArtItems` in the current document.

```
// This script counts all paragraphs in current document
// and reports the total.

if (documents.length > 0)
{
    numberOfParas = 0;

    for (i=0; i<activeDocument.textArtItems.length; i++)
   {
        curTextArt = activeDocument.textArtItems[i];
        curTextRange = curTextArt.textRange();
        numberOfParas = numberOfParas + curTextRange.paragraphs.length;
    }
    if (numberOfParas > 1)
   {
        alert("There are " + numberOfParas +
          " paragraphs in the document.");
    }
   else
   {
        alert("There is only one paragraph in the document.");
    }
}
```

PathItem

A path. A path is comprised of path points that define its geometry.

Properties

Property	R/O	Value type	What it is
area	R/O	Number	The area of this path in square points. An area may be negative or even 0. The paths winding order is determined by the sign of area. If the area is negative, the path is wound counter-clockwise. Self-intersecting paths may contain sub-areas that cancel each other out. Therefore, it is possible for a path's area to appear as zero even though it has apparent area.
artworkKnockout		KnockoutState constant	Is this object used to create a knockout? If so, what kind of knockout?
blendingMode		BlendModes constant	The mode used when compositing an object.
clipping		Boolean	Is this path to be used as a clipping path?
closed		Boolean	Is this path closed?
controlBounds	R/O	Array (of 4 numbers)	The bounds of the object including stroke width and controls.
editable	R/O	Boolean	Is this item editable?
evenodd		Boolean	Use the even-odd rule to determine insideness?
fillColor		Color object	The fill color of the path.
filled		Boolean	Should the path be filled?
fillOverprint		Boolean	Will art beneath a filled object be overprinted?
geometricBounds	R/O	Array (of 4 numbers)	The bounds of the object excluding stroke width.
guides		Boolean	Is this path a guide object?
height		Number	The height of the path item excluding stroke width.
hidden		Boolean	Is this path item hidden?
isIsolated		Boolean	Is this object isolated?

Property	R/O	Value type	What it is
layer	R/O	Layer object	The layer to which this path item belongs.
left		Number	The position of the left side of the item.
locked		Boolean	Is this path item locked?
name		String	The name of this path item.
note		String	The note text assigned to the path.
opacity		Number	The opacity of the object . The value is between 0.0 and 100.0.
parent	R/O	Layer object, GroupItem object, or CompoundPathItem object	The parent of this object.
pathPoints	R/O	PathPoints collection object	The path points contained in this path item.
polarity		PolarityValues constant	The polarity of the path.
position		Array (of 2 numbers)	The position of the top left corner of the path item excluding stroke width.
resolution	R/O	Number	The resolution of the path (in dots per inch).
selected		Boolean	Is this object selected?
selectedPathPoints	R/O	PathPoints collection object	All of the selected path points in the path.
sliced		Boolean	Is the item sliced? Default: false
strokeCap		StrokeCap constant	The type of line capping.
strokeColor		Color object	The stroke color for the path.
stroked		Boolean	Should the path be stroked?
strokeDashes		Array	Dash lengths. Set to an empty array for a solid line.
strokeDashOffset		Number	The default distance into the dash pattern at which the pattern should be started.
strokeJoin		StrokeJoin constant	Type of joints for the path.
strokeMiterLimit		Number	Are joins mitered (pointed) or beveled (squared-off)?
strokeOverprint		Boolean	Will art beneath a stroked object be overprinted?
strokeWidth		Number	Width of stroke.

Property	R/O	Value type	What it is
tags	R/O	Tags collection object	The tags contained in this path item.
top		Number	The position of the top of the item.
typename	R/O	String	Returns the name of the referenced object.
url		String	The value of the Adobe URL tag assigned to this path item.
visibilityVariable		Variable object	The visibility variable bound to the item.
visibleBounds	R/O	Array (of 4 numbers)	The visible bounds of the path item including stroke width.
width		Number	The width of the path item excluding stroke width.
zOrderPosition	R/O	Number	The position of this art object within the stacking order of the group or layer (Parent) that contains the art object.

Methods

Method	Parameter type	Returns	What it does
duplicate()	none	Item	Creates a duplicate of the selected item.
moveAfter(destination)	object	Nothing	Moves the item behind the specified object.
moveBefore(destination)	object	Nothing	Moves the item in front of the specified object.
moveToBeginning(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the front of the specified container.
moveToEnd(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the end of the specified container.
remove()	none	Nothing	Removes the referenced item from the document.

Method	Parameter type	Returns	What it does
resize(scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])	 number number boolean boolean boolean boolean boolean Transformation constant	Nothing	Scales the art object where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%.
rotate(angle [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,rotateAbout])	 number boolean boolean boolean boolean Transformation constant	Nothing	Rotates the art object relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
setEntirePath(pathPointArray)	Array	Nothing	Set the path using the array of anchor points. Each anchor point in the array is represented by its own array of 2 numbers.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.
transform transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])	 matrix object boolean boolean boolean boolean number Transformation constant	Nothing	Transforms the art object by applying a transformation matrix.
translate([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])	 number number boolean boolean boolean boolean	Nothing	Repositions the art object relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.

Method	Parameter type	Returns	What it does
zOrder(ZOrderMethod)	ZOrderMethod constant	Nothing	Arranges the art object's position in the stacking order of the group or layer (parent) of this object.

Notes

The `PathItem` class give you complete access to paths in Illustrator.

The `setEntirePath()` method provides an extremely efficient way to create paths comprised of straight lines.

Example 1

This script sets the stroke color and the fill color of the first path in the frontmost document.

```
// Example of how to set the stroke and fill of a PathItem
// this script assumes that there are at least 16 swatches.

if (documents.length > 0 && activeDocument.pathItems.length > 0)
{
   frontDocument = activeDocument;
   firstPath = frontDocument.pathItems[0];
   frontDocument.selection = firstPath;

   firstPath.filled = true;
   firstPath.fillColor = frontDocument.swatches[10].color;
   firstPath.stroked = true;
   firstPath.strokeWidth = 5;
   firstPath.strokeColor = frontDocument.swatches[15].color;
}
```

Example 2

This script illustrates the use of the `SetEntirePath()` method to create a new path consisting of straight lines.

```
// Example of how to create a new path consisting of 10 straight lines

if (documents.length > 0)
{
   var lineList = Array(10);
   for (index=0; index<10; index++)
   {
      lineList[index] = Array(index * 10 + 50, (index - 5) ^ 2 * 5 + 50);
   }

   frontDocument = activeDocument;
   newPath = frontDocument.pathItems.add();
   newPath.setEntirePath(lineList);
}
```

PathItems

A collection of paths.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add		PathItem object	Creates a new object.
ellipse([top] [,left] [,width] [,height] [,reversed] [,inscribed])	 number number number number boolean boolean	PathItem object	Creates a new pathItem in the shape of an ellipse using the supplied parameters.
polygon([centerX] [,centerY] [,radius] [,sides] [,reversed])	 number number number number boolean	PathItem object	Creates a new pathItem in the shape of an polygon using the supplied parameters.
rectangle([top] [,left] [,width] [,height] [,reversed])	 number number number number boolean	PathItem object	Creates a new pathItem in the shape of an polygon using the supplied parameters.
removeAll()		Nothing	Deletes all objects in this collection.

Method	Parameter type	Returns	What it does
roundedRectangle([top] [,left] [,width] [,height] [,horizontalRadius] [,verticalRadius] [,reversed])	 number number number number number number boolean	PathItem object	Creates a new pathItem in the shape of a rectangle with rounded corners using the supplied parameters.
star([centerX] [,centerY] [,radius] [,innerRadius] [,points] [,reversed])	 number number number number number boolean	PathItem object	Creates a new path item in the shape of a star using the supplied parameters.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

The methods `ellipse`, `polygon`, `rectangle`, `roundedRectangle`, and `star` allow you to create complex path items using straightforward parameters. If you do not provide any parameters when calling these methods, default values will be used.

Example

This script illustrates how to create a new rectangle in the first layer of the frontmost document.

```
// Example of how to create a rectangle in layer 1 of document 1

if (documents.length > 0)
{
   frontDocument = activeDocument;
   pathsInDocument = frontDocument.pathItems;

   // create a new rectangle with
   // top = 400, left = 50, witdth = 150 and height = 100

   newRectangle = pathsInDocument.rectangle(400, 50, 150, 100);
}
```

PathPoint

A point on a specific path. Each path point is made up of an anchor point (`anchor`) and a pair of handles (`leftDirection` and `rightDirection`).

Properties

Property	R/O	Value type	What it is
anchor		Array (of 2 numbers)	The position of this point's anchor point.
leftDirection		Array (of 2 numbers)	The position of this path point's in control point.
parent	R/O	PathItem object	The path item that contains this path point.
pointType		PointType constant	The type of path point, either a curve or a corner.
rightDirection		Array (of 2 numbers)	The position of this path point's out control point.
selected		PathPointSelection constant	Are points of this path point selected? If so, which one(s)?.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
remove()	none	Nothing	Removes the referenced point from the path.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

A `PathPoint` represents a point on a path, with its pair of control points, or handles. Any point can considered a corner point. Setting the `pointType` property of a path point to a corner forces the left and right direction points to be on a straight line when the user attempts to modify them in the user interface.

PathPoints

A collection of path points in a specific path.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	PathPoint object	Creates a new PathPoint object.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This script illustrates how to add a new path point to an existing path.

```
// Example of how to add a new PathPoint to an existing path

if (documents.length > 0 && activeDocument.pathItems.length > 0)
{
    firstPath = activeDocument.pathItems[0];
    newPoint = firstPath.pathPoints.add();

    newPoint.anchor = Array(75, 300);
    newPoint.leftDirection = Array(10, 280);
    newPoint.rightDirection = Array(165, 330);
    newPoint.pointType = PointType.CORNER;
}
```

Pattern

A pattern definition contained in a document.

Properties

Property	R/O	Value type	What it is
name	R/O	String	The pattern name.
parent	R/O	document object	The document that contains this pattern.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
remove()	none	Nothing	Removes the referenced pattern from the document.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

The `Pattern` object represents a pattern as defined in the Illustrator document.

PatternColor

A pattern color specification, used in conjunction with the `Pattern` property of the `Color` specification.

Properties

Property	R/O	Value type	What it is
matrix		Matrix object	An additional transformation matrix to manipulate the prototype pattern,
pattern		Pattern object	A reference to the pattern object that defines the pattern to use in this color definition.
reflect		Boolean	Is the prototype reflected before filling?
reflectAngle		Number	The axis (in degrees) around which to reflect.
rotation		Number	The angle (in degrees) to rotate the prototype pattern before filling.
scaleFactor		Array (of 2 numbers)	The fraction to scale the prototype pattern before filling, represented as point containing horizontal and vertical scaling percentages.
shearAngle		Number	The angle (in degrees) to slant the shear by.
shearAxis		Number	The axis (in degrees) to shear with respect to.
shiftAngle		Number	The angle (in degrees) to translate the unscaled prototype pattern before filling
shiftDistance		Number	The distance to translate the unscaled prototype pattern before filling.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Pattern colors are created using a reference to an existing pattern in the document. A matrix may be specified to further transform the pattern color.

Example

This script illustrates how to modify the first pattern in a document.

```
// Example of how to change the pattern in the frontmost
// pattern in a document and of the pattern in the palette.


if (documents.length > 0 && activeDocument.pathItems.length > 0)
{
  for (i = 0; i < activeDocument.swatches.length; i++)
  {
    // Get the generic color object of the swatch
    currentSwatch = activeDocument.swatches[i];
    swatchColor = currentSwatch.color;


    // Only operate on patterns
    if (swatchColor.color == ColorType.PATTERN)
    {
      // Obtain the PatternColor from generic color object
      colorOfPattern = swatchColor.pattern;

      // Change the pattern properties
      colorOfPattern.rotation = 10;

      // Set the PatternColor of the original Color object
      swatchColor.pattern = colorOfPattern;

      // Apply the color to the frontmost path
      firstPath = activeDocument.pathItems[0];
      firstPath.filled = true;
      firstPath.fillColor = swatchColor;

      // Change the definition of the pattern in the palette
      //swatchRef.color = swatchColor;
      currentSwatch.color = swatchColor;
    }
  }
}
```

Patterns

A collection of patterns in a document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	Pattern object	Creates a new object.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This script illustrates how to remove a pattern. Note after removing Illustrator objects you should set the variable that referenced the object you just removed to Nothing.

```
// Example of how to remove the second pattern in a document.
// Note: Set the patternToRemove reference to nothing
// because it no longer references an existing
// Illustrator pattern

if (documents.length > 0)
{
    frontDocument = activeDocument;
    patternToRemove = frontDocument.patterns[1];
    patternToRemove.remove();
    patternToRemove = null;
}
```

PDFOpenOptions

Options you can specify when opening a PDF file. See the open method in the Application object for additional details.

Properties

Property	R/O	Value type	What it is
pageToOpen		Number	What page should be displayed when opening a multipage document. Default: 1
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

This object is used to specify which page you wish to access when opening a multipage PDF document. PDFOpenOptions can only be supplied in the open method.

You do not have to specify values for this property; Illustrator will assign it the default value of 1.

PDFSaveOptions

Options which may be supplied when saving a document as an Acrobat PDF file. See the `Save` method for additional details.

Properties

Property	R/O	Value type	What it is
colorCompression		CompressionQuality constant	The type of color bitmap compression used.
colorDownsampling		Number	The color downsampling resolution in dots per inch (dpi). If the value is 0, no downsampling is performed.
compatibility		PDFCompatibility constant	Specifies the version of the Acrobat file format to create.
compressArt		Boolean	Is line art and text to be compressed?
embedAllFonts		Boolean	Are all fonts to be embedded?
embedICCProfile		Boolean	Should a ICC profile be embedded in the saved file?
fontSubsetThreshold		Number	Include a subset of fonts when less than this percentage of characters is used in the document. Valid for Illustrator 9 file format.
generateThumbnails		Boolean	Should thumbnail images be generated with the saved file?
grayscaleCompression		CompressionQuality constant	Quality of grayscale bitmap compression.
grayscaleDownsampling		Number	Downsampling resolution in dots per inch (dpi). If the value is 0, no downsampling is performed.
monochromeCompression		MonochromeCompression constant	Specifies type of monochrome bitmap compression used.
monochromeDownsampling		Number	Downsampling resolution in dots per inch (dpi). If the value is 0, no downsampling is performed.
preserveEditability		Boolean	Should Illustrator editing capabilities be preserved when saving the document?
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

PDF save options can only be supplied in conjunction with the `saveAs` method.

It is not necessary to specify values for all properties. Default values will be provided for any properties not specified.

Example

This script illustrates how to save the frontmost document as PDF.

```
// This script shows how to save the current document as PDF

if (documents.length > 0)
{
   documentPath = activeDocument.path + "/" + activeDocument.name;
   theFile = new File(documentPath);

   thePDFSaveOptions = new PDFSaveOptions();
   documents[0].saveAs(theFile, thePDFSaveOptions);
}
```

PhotoshopFileOptions

Properties

Property	R/O	Value type	What it is
parent	R/O	Object	The parent of this object.
preserveImageMaps		Boolean	Preserve image maps when document is converted? Default: true
preserveLayers		Boolean	Preserve layers when document is converted? Default: true
preserveSlices		Boolean	Preserve slices when document is converted? Default: true
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

PlacedItem

An artwork item (optionally stored in an external file) placed in a document. A placed item must correspond to a file containing vector-graphic data, such as a PICT, EPS or PDF file.

Properties

Property	R/O	Value type	What it is
artworkKnockout		KnockoutState constant	Is this object used to create a knockout? If so, what kind of knockout? You cannot set this value to knockoutUnknown.
blendingMode		BlendModes constant	The mode used when compositing an object.
boundingBox		Array (of 4 numbers)	The dimensions of the placed art object regardless of transformations.
contentVariable		Variant	The content variable bound to the item.
controlBounds	R/O	Array (of 4 numbers)	The bounds of the object including stroke width and controls.
editable	R/O	Boolean	Is this item editable?
file		File object	The file containing the artwork.
geometricBounds	R/O	Array (of 4 numbers)	The bounds of the object excluding stroke width.
height		Number	The height of the placed artwork.
hidden		Boolean	Is this item hidden?
isIsolated		Boolean	Is this object isolated?
layer	R/O	Layer object	The layer to which this item belongs.
left		Number	The position of the left side of the item.
locked		Boolean	Is this item locked?
matrix		Matrix	The transformation matrix of the placed artwork.
name		String	The name of this item.
opacity		Number	The opacity of the object . The value is between 0.0 and 100.0.
parent	R/O	Layer object or GroupItem object	The parent of this object.
position		Array (of 2 numbers)	The position of the top left corner of the item.

Property	R/O	Value type	What it is
selected		Boolean	Is this object selected?
sliced		Boolean	Is the item sliced? Default: false
tags	R/O	Tags collection object	The tags contained in this item.
top		Number	The position of the top of the item.
typename	R/O	String	Returns the name of the referenced object.
url		String	The value of the Adobe URL tag assigned to this item.
visibilityVariable		Variable object	The visibility variable bound to the item.
visibleBounds	R/O	Array (of 4 numbers)	The visible bounds of the item including stroke width.
width		Number	The width of the item.
zOrderPosition	R/O	Number	The position of this art object within the stacking order of the group or layer (parent) that contains the art object.

Methods

Method	Parameter type	Returns	What it does
duplicate()	none	Item	Creates a duplicate of the selected item.
moveAfter(destination)	object	Nothing	Moves the item behind the specified object.
moveBefore(destination)	object	Nothing	Moves the item in front of the specified object.
moveToBeginning(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the front of the specified container.
moveToEnd(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the end of the specified container.
remove()	none	Nothing	Removes the referenced item from the document.

Method	Parameter type	Returns	What it does
resize(scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])	 number number boolean boolean boolean boolean boolean Transformation constant	Nothing	Scales the art object where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%.
rotate(angle [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,rotateAbout])	 number boolean boolean boolean boolean Transformation constant	Nothing	Rotates the art object relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.
transform transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])	 matrix object boolean boolean boolean boolean number Transformation constant	Nothing	Transforms the art object by applying a transformation matrix.
translate([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])	 number number boolean boolean boolean boolean	Nothing	Repositions the art object relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
zOrder(ZOrderMethod)	ZOrderMethod constant	Nothing	Arranges the art object’s position in the stacking order of the group or layer (parent) of this object.

Notes

When you create a placed item, Illustrator may display a dialog. To avoid this dialog check the box to turn the warning off the first time the dialog is displayed.

Vector art files, such as EPS and PDF files, can be placed by users with the File > Place... command in Illustrator.

Example

This script illustrates how to change the selection of placed items.

```
// This script toggles the selection state of all placed items.
// If it is selected, it becomes deselected and if it is not selected
// it gets selected.

for (i=0; i < activeDocument.placedItems.length; i++)
{
   placedArt = activeDocument.placedItems[i];
   placedArt.selected = !(placedArt.selected);
}
```

PlacedItems

The collection of placed art items in the document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
add()	none	PlacedItem object	Creates a new object.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

See example under PlacedItem for sample script using the PlacedItems collection object.

PluginItem

An art object created by an Illustrator plug-in.

Properties

Property	R/O	Value type	What it is
artworkKnockout		KnockoutState constant	Is this object used to create a knockout? If so, what kind of knockout? You cannot set this value to knockoutUnknown.
blendingMode		BlendModes constant	The mode used when compositing an object.
controlBounds	R/O	Array (of 4 numbers)	The bounds of the object including stroke width and controls.
editable	R/O	Boolean	Is this item editable?
geometricBounds	R/O	Array (of 4 numbers)	The bounds of the object excluding stroke width.
height		Number	The height of the page item.
hidden		Boolean	Is this item hidden?
isIsolated		Boolean	Is this object isolated?
layer	R/O	Layer object	The layer to which this item belongs.
left		Number	The position of the left side of the item.
locked		Boolean	Is this item locked?
name		String	The name of this item.
opacity		Number	The opacity of the object . The value is between 0.0 and 100.0.
parent	R/O	Layer object or GroupItem object	The parent of this object.
position		Array (of 2 numbers)	The position of the top left corner of the item.
selected		Boolean	Is this object selected?
sliced		Boolean	Is the item sliced? Default: false
tags	R/O	Tags collection object	The tags contained in this item.
top		Number	The position of the top of the item.
typename	R/O	String	Returns the name of the referenced object.

Property	R/O	Value type	What it is
url		String	The value of the Adobe URL tag assigned to this item.
visibilityVariable		Variable object	The visibility variable bound to the item.
visibleBounds	R/O	Array (of 4 numbers)	The visible bounds of the item including stroke width.
width		Number	The width of the item.
zOrderPosition	R/O	Number	The position of this art object within the stacking order of the group or layer (parent) that contains the art object.

Methods

Method	Parameter type	Returns	What it does
duplicate()	none	Item	Creates a duplicate of the selected item.
moveAfter(destination)	object	Nothing	Moves the item behind the specified object.
moveBefore(destination)	object	Nothing	Moves the item in front of the specified object.
moveToBeginning(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the front of the specified container.
moveToEnd(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the end of the specified container.
remove()	none	Nothing	Removes the referenced item from the document.
resize(scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])	number number boolean boolean boolean boolean boolean transformation constant	Nothing	Scales the art object where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%.

Method	Parameter type	Returns	What it does
rotate(angle [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,rotateAbout])	 number boolean boolean boolean boolean transformation constant	Nothing	Rotates the art object relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.
transform transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])	 matrix object boolean boolean boolean boolean number transformation constant	Nothing	Transforms the art object by applying a transformation matrix.
translate([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])	 number number boolean boolean boolean boolean	Nothing	Repositions the art object relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
zOrder(ZOrderMethod)	ZOrderMethod constant	Nothing	Arranges the art object's position in the stacking order of the group or layer (parent) of this object.

Notes

Plug-in items cannot be created from a script. You may copy plug-in items by using the duplicate() method and then using the one of the move methods (moveAfter(), moveBefore(), moveToBeginning(), and moveToEnd()), to place the item at the proper location.

Example

This example demonstrates how to create a new plugin item by copying an existing `pluginItem`.

```
// Example of how to create Plug-in art by copying existing plugin art items

if (documents.length > 0 && activeDocument.pluginItems.length > 0)
{
   thePluginArt = activeDocument.pluginItems[0];
   thePluginArt.duplicate();
   thePluginArt.moveToBeginning(activeDocument);
}
```

PluginItems

A collection of plugin items in a document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Plugin items cannot be created from a script. You may copy plug-in items by using the duplicate() method and then using the one of the move methods (moveAfter(), moveBefore(), moveToBeginning(), and moveToEnd()), to place the item at the proper location.

Example

See example under PluginItem for an example of how to use the PluginItems collection.

Preferences

Properties

Property	R/O	Value type	What it is
parent	R/O	Object	The parent of this object.
photoshopFileOptions	R/O	OpenOptionsPhotoshop object	Options to use when opening or placing a Photoshop file.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

RasterItem

A bitmap art object in a document.

Properties

Property	R/O	Value type	What it is
artworkKnockout		KnockoutState constant	Is this object used to create a knockout? If so, what kind of knockout? You cannot set this value to knockoutUnknown.
blendingMode		BlendModes constant	The mode used when compositing an object.
boundingBox		Array (of 4 numbers)	Dimensions of the raster art object regardless of transformations.
contentVariable		Variant	The content variable bound to the item.
controlBounds	R/O	Array (of 4 numbers)	The bounds of the object including stroke width and controls.
editable	R/O	Boolean	Is this item editable?
embedded		Boolean	Is the raster art object embedded in the Illustration?
file		File object	The file containing the raster art work.
geometricBounds	R/O	Array (of 4 numbers)	The bounds of the object excluding stroke width.
height		Number	The height of the page item.
hidden		Boolean	Is this item hidden?
imageColorSpace		ImageColorSpace object	The color space of the raster image.
isIsolated		Boolean	Is this object isolated?
layer	R/O	Layer object	The layer to which this item belongs.
left		Number	The position of the left side of the item.
locked		Boolean	Is this item locked?
matrix		Matrix object	The transformation matrix of the raster art object.
name		String	The name of this item.
opacity		Number	The opacity of the object . The value is between 0.0 and 100.0.

Property	R/O	Value type	What it is
parent	R/O	Layer object or GroupItem object	The parent of this object.
position		Array (of 2 numbers)	The position of the top left corner of the item.
selected		Boolean	Is this object selected?
sliced		Boolean	Is the item sliced? Default: false
status		rasterLinkState constant	Status of the linked image.
tags	R/O	Tags collection object	The tags contained in this item.
top		Number	The position of the top of the item.
typename	R/O	String	Returns the name of the referenced object.
url		String	The value of the Adobe URL tag assigned to this item.
visibilityVariable		Variable object	The visibility variable bound to the item.
visibleBounds	R/O	Array (of 4 numbers)	The visible bounds of the item including stroke width.
width		Number	The width of the item.
zOrderPosition	R/O	Number	The position of this art object within the stacking order of the group or layer (parent) that contains the art object.

Methods

Method	Parameter type	Returns	What it does
colorize(color)	Color object	Nothing	Colorize the raster item with a CMYK or RGB Color.
duplicate()	none	Item	Creates a duplicate of the selected item.
moveAfter(destination)	object	Nothing	Moves the item behind the specified object.
moveBefore(destination)	object	Nothing	Moves the item in front of the specified object.
moveToBeginning(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the front of the specified container.

Method	Parameter type	Returns	What it does
moveToEnd(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the end of the specified container.
remove()	none	Nothing	Removes the referenced item from the document.
resize(scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])	 number number boolean boolean boolean boolean boolean transformation constant	Nothing	Scales the art object where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%.
rotate(angle [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,rotateAbout])	 number boolean boolean boolean boolean transformation constant	Nothing	Rotates the art object relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.
transform transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])	 matrix object boolean boolean boolean boolean number transformation constant	Nothing	Transforms the art object by applying a transformation matrix.
translate([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])	 number number boolean boolean boolean boolean	Nothing	Repositions the art object relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
zOrder(ZOrderMethod)	ZOrderMethod constant	Nothing	Arranges the art object’s position in the stacking order of the group or layer (parent) of this object.

Notes

Raster items can be created from a script if an external file is used. You can create new raster items by using the duplicate() method with an existing item and then moving it to the desired location with one of the move methods — moveAfter(), moveBefore(), moveToBeginning(), and moveToEnd().

Example

This example illustrates how to create a new raster item in the frontmost document. The script assumes that you have a file called "/temp/sample.jpg".

```
// Example of how to create a new RasterItem in the frontmost
// document. This script assumes that you have a sample file
// at //temp/sample.jpg

if (documents.length == 0)
{
   documents.add();
}

rasterItemFile = new File("//temp/sample.jpg");

newRasterArt = activeDocument.rasterItems.add();
newRasterArt.file = rasterItemFile;
newRasterArt.position = Array(100, 400);
```

RasterItems

A collection of raster art items.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	RasterItem object	Creates a new object.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This script illustrates how to obtain the color space of a raster item.

```
// This script examines the color space of the first raster item
// in the document

if (documents.length > 0 && activeDocument.rasterItems.length > 0)
{
   theRasterArt = activeDocument.rasterItems[0];

   switch (theRasterArt.imageColorSpace)
   {
      case ImageColorSpace.CMYK:
      alert("The first raster item is a CMYK raster item");
      break;

      case ImageColorSpace.RGB:
      alert("The first raster item is an RGB raster item");
      break;

      case ImageColorSpace.GRAYSCALE:
      alert("The first raster item is a Grayscale raster item");
      break;
   }
}
```

RGBColor

A RGB color specification, used in conjunction with the `RGB` property of the `Color` specification.

Properties

Property	R/O	Value type	What it is
blue		Number	The blue color value as a value in the range 0.0 - 255.0.
green		Number	The green color value as a value in the range 0.0 - 255.0.
red		Number	The red color value as a value in the range 0.0 - 255.0.
typename	R/O	String	Returns the name of the referenced object.

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

If the `DocumentColorSpace` of a document is `CMYKColor` and you specify the color value for a page item in that document using `RGBColor`, Illustrator will translate the RGB color specification into a CMYK color specification. The same thing happens if the document's `DocumentColorSpace` is `RGBColor` and you specify colors using `CMYKColor`. Since this translation can cause information loss you should specify colors using the class that matches the document's `documentColorSpace`.

Example

This script sets the default fill color of the frontmost document to yellow using an RGB object. If the color space of the frontmost document is CMYK, then Illustrator will regard the RGB fill color as a CMYK color although it is specified using RGB.

```
// This script sets the default fill color to yellow.
// If the color space is CMYK then Illustrator
// automatically translates the RGB color to its CMYK equivalence

if (documents.length > 0)
{
   // Define the new color
   newRGBColor = new RGBColor();
   newFillColor = new Color();

    newRGBColor.red = 255;
    newRGBColor.green = 255;
    newRGBColor.blue = 0;

   // Wrap the RGB color in a generic color object
   // and set that as the default fill color

    newFillColor.rgb = newRGBColor;
    activeDocument.defaultFillColor = newFillColor;
}
```

Spot

A spot color definition contained in the Illustrator document.

Properties

Property	R/O	Value type	What it is
color		Color object	The color information for this spot color.
colorType		ColorModel constant	Color model of the spot color.
name		String	The spot color’s name.
parent	R/O	Document object	The document that contains this spot color.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
remove()	none	Nothing	Removes the referenced item from the document.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Illustrator’s `Spot` object represents a spot color as defined by Illustrator. All Illustrator documents contain the spot color "[Registration]" which can be used to print to all plates of a separation.

If no properties are specified when creating a new spot, default properties will be provided. However, if specifying the color, you must use the same color space as the document, either CMYK or RGB. Otherwise, an error will result. When created, the spot is inserted into the swatch palette at the end.

Example

This script illustrates how to create a new spot in the frontmost document.

```
// Example of creating a new spot in the frontmost document

if (documents.length > 0)
{
   newRGBColor = new RGBColor();
   newColor = new Color();

   // Define the new color value
   newRGBColor.red = 255;
   newRGBColor.green = 0;
   newRGBColor.blue = 0;

   newColor.rgb = new RGBColor();

   // Create the new spot
   frontDocument = activeDocument;
   newSpot = frontDocument.spots.add();

   // Define the new SpotColor as 80% of the specified RGB color
   newSpot.name = "My New Red spot color";
   newSpot.tint = 80;
   newSpot.color = newColor;
}
```

SpotColor

A spot color specification, used in conjunction with the `spot` property of the `color` specification.

Properties

Property	R/O	Value type	What it is
spot		Spot object	A reference to the spot color object which defines the color.
tint		Number	The tint of the color as a value in the range 0.0 - 100.0.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Spot colors are specified using a numeric value that ranges from 0.0 to 100.0 for the tint of the color. The `color` property must be set to a reference to an existing spot color.

Spots

A collection of spot colors in a document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	Spot object	Creates a new object.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example 1

This script illustrates how to remove all spots defined in the frontmost document.

```
// Example of how to remove all spots from
// the frontmost document

if (documents.length > 0)
{
   documentSpots = activeDocument.spots;
   documentSpots.removeAll();
}
```

Example 2

This script shows how to create a new spot, and they applying a tint of that spot to the fill of a path item.

```
// Example of how to define and apply a spot color

if (documents.length > 0 && activeDocument.pathItems.length > 0)
{
   // Define the new color value
   newRGBColor = new RGBColor();
   newColor = new Color();

   newRGBColor.red = 255;
   newRGBColor.green = 0;
   newRGBColor.blue = 0;

   newColor.rgb = newRGBColor;

   // Create the new spot
   frontDocument = activeDocument;
   newSpot = frontDocument.spots.add();

   // Define the new SpotColor as 80% of
   // the specified RGB color
   newSpot.name = "Red spot color";
   newSpot.tint = 80;
   newSpot.color = newColor;

   // Now apply a 50% of the spot color we just created
   // to the frontmost path item. We do this by creating
   // a spotcolor object and setting the specifications
   // on that object. We then wrap the spot color object
   // in a generic color object and use it to set the fill
   // color for the first path item in the frontmost document

   newSpotColor = new SpotColor();
   newPathColor = new Color();

   newSpotColor.spot = newSpot;
   newSpotColor.tint = 50;
   newPathColor.spot = newSpotColor;

   frontPath = frontDocument.pathItems[0];
   frontPath.filled = true;
   frontPath.fillColor = newPathColor;
}
```

Swatch

A color swatch definition contained in a document.

Properties

Property	R/O	Value type	What it is
color		Color object	The color information for this swatch.
name		String	The swatch's name.
parent	R/O	Document object	The document that contains this swatch.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
remove()	none	Nothing	Removes the referenced item from the document.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

The swatches correspond to the swatch palette in Illustrator's user interface. Additional swatches can be created either manually by a user or by a script. The swatch can hold all types of color data (i.e., pattern, gradient, CMYK, RGB, gray, spot).

Example

This script illustrates how to change the name of the fifth swatch.

```
// Example of how to change the name of the fifth swatch

if (documents.length > 0 && activeDocument.swatches.length >= 5)
{
   fifthSwatch = activeDocument.swatches[4];
   fifthSwatch.name = "ThisIsThe5thSwatch";
}
```

Swatches

A collection of swatches in a document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	Swatch object	Creates a new Swatch object.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This script illustrates how to first obtain a swatch by name and then how to delete that swatch.

```
// Example of how to remove the swatch called
// "Orange M=50 Y=100" in the frontmost document

if (documents.length > 0)
{
    swatchToDelete = activeDocument.swatches["Orange M=50 Y=100"];
    swatchToDelete.remove();
}
```

Symbol

A `symbol` is an `artObject` that is stored in the Symbols Palette and can be reused one or more times in the document without duplicating the art data. `Symbols` are contained in documents.

Properties

Property	R/O	Value type	What it is
name		String	The symbol's name.
parent		Object	The object that contains the symbol object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
duplicate()	none	Symbol object	Returns a duplicate of the selected object.
remove()	none	Nothing	Removes the specified and returns the object that was removed.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

SymbolItem

A `SymbolItem` is an instance of a symbol in a document. `SymbolItems` are linked to the symbols from which they were created and change with any modification of those symbols.

Properties

Property	R/O	Value type	What it is
artworkKnockout		KnockoutState constant	Is this object used to create a knockout? If so, what kind of knockout? You cannot set this value to knockoutUnknown.
blendingMode		BlendModes constant	The mode used when compositing an object.
controlBounds	R/O	Array (of 4 numbers)	The bounds of the object including stroke width and controls.
editable	R/O	Boolean	Is this symbolItem editable?
geometricBounds	R/O	Array (of 4 numbers)	The bounds of the object excluding stroke width.
height		Number	The height of the symbol item.
hidden		Boolean	Is this symbolItem hidden?
isIsolated		Boolean	Is this object isolated?
layer	R/O	Layer object	The layer to which this symbolItem belongs.
left		Number	The position of the left side of the symbolItem.
locked		Boolean	Is this symbolItem locked?
name		String	The name of this symbolItem.
opacity		Numbers	The opacity of the object . The value is between 0.0 and 100.0.
parent	R/O	Layer object or GroupItem object	The parent of this object.
position		Array (of 2 numbers)	The position of the top left corner of the symbolItem.
selected		Boolean	Is this object selected?
sliced		Boolean	Is the symbolItem sliced? Default: false
symbol		SymbolObject	The symbol that was used to create this symbol item.
tags	R/O	Tags collection object	The tags contained in this symbolItem.

Property	R/O	Value type	What it is
top		Number	The position of the top of the symbolItem.
typename	R/O	String	Returns the name of the referenced object.
url		String	The value of the Adobe URL tag assigned to this symbolItem.
visibilityVariable		Variable object	The visibility variable bound to the symbolItem.
visibleBounds	R/O	Array (of 4 numbers)	The visible bounds of the symbolItem including stroke width.
width		Number	The width of the symbolItem.
zOrderPosition	R/O	Number	The position of this art object within the stacking order of the group or layer (parent) that contains the art object.

Methods

Method	Parameter type	Returns	What it does
duplicate()	none	SymbolItem	Create a duplicate of the selected symbolItem.
moveAfter(destination)	object	Nothing	Moves the symbolItem behind the specified object.
moveBefore(destination)	object	Nothing	Moves the symbolItem in front of the specified object.
moveToBeginning (destination)	object (document, layer, or groupItem)	Nothing	Moves the symbolItem to the front of the specified container.
moveToEnd(destination)	object (document, layer, or groupItem)	Nothing	Moves the symbolItem to the end of the specified container.
remove()	none	Nothing	Removes the symbolItem from the document.
resize(scaleX, scaleY [,changePositions], [,changeFillPatterns], [,changeFillGradients], [,changeStrokePattern], [,changeLineWidths] [,scaleAbout])	number boolean boolean boolean boolean boolean transformation constant	Nothing	Scales the art object where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%.

Method	Parameter type	Returns	What it does
rotate(Angle [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,rotateAbout])	number boolean boolean boolean boolean transformation constant	Nothing	Rotates the art object relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.
transform (transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])	 matrix object boolean boolean boolean boolean number transformation constant	Nothing	Transforms the art object by applying a transformation matrix.
translate([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])	number boolean boolean boolean boolean	Nothing	Repositions the art object relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
zOrder(ZOrderMethod)	ZOrderMethod constant	Nothing	Arranges the art object’s position in the stacking order of the group or layer (Parent) of this object.

Notes

The `moveAfter()` and `moveBefore()` methods do not change the position of the object on the art board. They change the order in which Illustrator draws the objects and the containment hierarchy.

The `moveToBeginning()` and `moveToEnd()` methods place the object in the specified container ahead of or behind other objects, respectively.

SymbolItems

The collection of symbol items in the document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add(symbol)	Symbol	SymbolItem	Creates an instance of the specified symbol.
removeAll()	none	Nothing	Deletes all objects in the collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Symbols

The collection of symbols in the document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of symbolObjects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add(sourceArt)	Art object	Symbol object	Returns a Symbol object created from the source art object.
removeAll()	none	Nothing	Deletes all objects in the collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

You can create a new symbol from any of the following source art classes:

- CompoundPathItems
- GraphItems
- GroupItems
- MeshItems
- PathItems
- PlacedItems
- TextPath_PathItems
- RasterItems
- SymbolItems
- TextArtItems

Tag

A label associated with a specific piece of artwork.

Properties

Property	R/O	Value type	What it is
name		String	The tag’s name.
parent	R/O	Object	The object that contains this tag.
typename	R/O	String	Returns the name of the referenced object.
value		String	The data stored in this tag.

Method	Parameter type	Returns	What it does
remove()	none	Nothing	Removes the referenced item from the document.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Tags allows you to assign an unlimited number of key-value pairs to any page item in a document.

Example

This example illustrates how to list the tags associated with the first selected item. The name and value of the tags are listed in a new document.

```
// The following example shows the tags of the selected art item
// the tags are shown in a separate document

selection = activeDocument.selection;

if ((selection.length > 0) && (selection instanceof Array))
{
   for (i = 0; i < selection.length; i++)
   {
      selectedArt = selection[0];
      tagList = selectedArt.tags;

      if (tagList.length == 0)
      {
         alert("The selected art has no tags");
      }
      else
      {
         // Create a document and add a line of text per tag
         reportDocument = documents.add();
         top_offset = 400;

         for (i = 0; i < tagList.length; i++)
         {
            tagText = tagList[i].value;
            newItem = reportDocument.textArtItems.add();
            newItem.contents = "Tag: (" + tagList[i].name +
                                    " , " + tagText + ")";
            newItem.position = Array(100, top_offset);
            top_offset = top_offset - 20
         }
      }
   }
}
else
{
   alert("No art items selected.");
}
```

Tags

A collection of tags.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	Tag object	Creates a new Tag object.
removeAll	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This example illustrates how to set the URL of all images in a document. It uses the special tag name “AdobeURL” to store the value of the URL.

```
// This example shows how to set the URL property on all
// RasterItem and all PlacedItems in the document

frontDocument = activeDocument;

for (i=0; i < frontDocument.pageItems.length; i++)
{
   imageArt = frontDocument.pageItems[i];

   alert(imageArt.typename);

   if ((imageArt.typename == "PlacedItem") ||
      (imageArt.typename == "RasterItem"))
   {

      // Create a new Tag with the name AdobeURL and the
      // value of the www link

      urlTAG = imageArt.tags.add();
      urlTAG.name = "AdobeWebSite";
      urlTAG.value = "http://www.adobe.com/";
   }
}
```

TextArtItem

A text art object or objects. From the user interface, this is text created with the Text tool

Properties

Property	R/O	Value type	What it is
artworkKnockout		KnockoutState constant	Is this object used to create a knockout? If so, what kind of knockout?
blendingMode		BlendModes constant	The mode used when compositing an object.
contents		String	The textual contents of the text art item.
contentVariable		Variable object	The content variable bound to the item.
controlBounds	R/O	Array (of 4 numbers)	The bounds of the object including stroke width and controls.
editable	R/O	Boolean	Is this item editable?
geometricBounds	R/O	Array (of 4 numbers)	The bounds of the object excluding stroke width.
height		Number	The height of the TextArtItem. You should only set the height of a TextArtItem that contains text.
hidden		Boolean	Is this text art item hidden?
isIsolated		Boolean	Is this object isolated?
kind		textType constant	The type of text art displayed by this object.
layer	R/O	Layer object	The layer to which this text art item belongs.
left		Number	The position of the left side of the item.
locked		Boolean	Is this text art item locked?
name		String	The name of this text art item.
opacity		Number	The opacity of the object . The value is between 0.0 and 100.0.
parent	R/O	Layer object or GroupItem object	The parent of this object.
position		Array (of 2 numbers)	The position of the top left corner of the text art item.
selected		Boolean	Is this text art item selected?

Property	R/O	Value type	What it is
selection		TextRange collection object	The selected text in the contents of this text art item.
sliced		Boolean	Is the item sliced? Default: false
tags	R/O	Tags collection object	The tags contained in this text art item.
textPath_PathItems	R/O	PathItems collection object	The path items associated with in-path and on-path text.
textPaths	R/O	TextPaths collection object	The text paths contained in this text art item,
top		Number	The position of the top of the item.
typename	R/O	String	Returns the name of the referenced object.
url		String	The value of the Adobe URL tag assigned to this text art item.
visibilityVariable		Variable object	The visibility variable bound to the item.
visibleBounds	R/O	Array (of 4 numbers)	The visible bounds of the text art item including stroke width.
width		Number	The width of the text art item. You should only try to set the width of a textartitem that contains text.
wrapped		Boolean	Does the text wrap around other objects (valid only for area text)?
zOrderPosition	R/O	Number	The position of this art object within the stacking order of the group or layer (Parent) that contains the art object.

Methods

Method	Parameter type	Returns	What it does
createOutline()	none	GroupItem object	Converts a text art item into a group item consisting of paths and compound paths.
duplicate()	none	item	Creates a duplicate of the selected item.
moveAfter(destination)	object	Nothing	Moves the item behind the specified object.

Method	Parameter type	Returns	What it does
moveBefore(destination)	object	Nothing	Moves the item in front of the specified object.
moveToBeginning(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the front of the specified container.
moveToEnd(destination)	object (document, layer, or groupItem)	Nothing	Moves the item to the end of the specified container.
remove()	none	Nothing	Removes the referenced item from the document.
resize(scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])	 number number boolean boolean boolean boolean boolean transformation constant	Nothing	Scales the art object where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%.
rotate(angle, [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,rotateAbout])	 number boolean boolean boolean boolean transformation constant	Nothing	Rotates the art object relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
textRange([,rangeStart] [,rangeEnd])	 number number	TextRange object	Returns a text range object referencing a substring of the current text range, where rangeStart is the beginning character position and rangeEnd is the ending position. The first character position is zero. If omitted, rangeStart defaults to 0. If omitted, rangeEnd defaults to the last character of the range.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Method	Parameter type	Returns	What it does
transform(transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])	 matrix object boolean boolean boolean boolean number Transformation constant	Nothing	Transforms the art object by applying a transformation matrix.
translate([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])	 number number boolean boolean boolean boolean	Nothing	Repositions the art object relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
zOrder(ZOrderMethod)	ZOrderMethod constant	Nothing	Arranges the art object's position in the stacking order of the group or layer (Parent) of this object.

Notes

There are three types of text art in Illustrator, as specified by the text art item's `kind` property. See Chapter 3 for more information on working with the three kinds of text art items.

Example

This example illustrates how to create a series of rotated text art items from a selected text art item. Before running this script you should create and select a text art item in Illustrator.

The example also illustrates how you can use the parent property of an objects to make sure that new objects are put into the same layer or group as the original item.

```
// This example shows how to rotate the selected text art item
// First check the selection of the application. It has to be
// a text art item for this script to run


if (documents.length > 0)
{
   selectedItems = activeDocument.selection;

   // check to make sure something is selected.
   if (selectedItems.length != 0)
   {
      pageObject = selectedItems[0];
      pageItemType = pageObject.typename;

      if (pageItemType == "TextArtItem")
      {
         // Get the parent of the text art so new text art items
         // can be inserted in the same group or layer with the
         // selected text art.

         textArtGroup = pageObject.parent.textArtItems;

         // Create 5 new versions of the text art each rotated a bit

         for (i=1; i<=5; i++)
         {
            newTextArt = textArtGroup.add();
            newTextArt.position = pageObject.position;
            newTextArt.contents = pageObject.contents;
            newTextArt.rotate(180 * i/6);
         }
      }
   }
}
```

TextArtItems

A collection of text art items.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
add()	none	TextArtItem object	Creates a new object.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

See the example under TextArtItem for a script that uses the TextArtItems collection.

TextFace

A text face (currently available font) in the document.

Properties

Property	R/O	Value type	What it is
name	R/O	String	The text face's name.
parent	R/O	Illustrator Application object	The application that contains this text face.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

The TextFace object provides access to the name of every font currently available to the Illustrator application.

Example

The following script illustrates how to apply the first text face to all text art in the frontmost document.

```
// Example of how to set the font of all the text in the document
// to the first text face

// Get the first text face in the document

if (documents.length > 0)
{
   fontToApply = textFaces[0];

   // Iterate through all text art and apply the font

   for (i=0; i< activeDocument.textArtItems.length; i++)
   {
      textArtRange = activeDocument.textArtItems[i].textRange();
      textArtRange.font = fontToApply.name;
   }
}
```

TextFaces

A collection of text faces (currently available font) in the document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Application object	The parent of this object.
typename	R/O	String	Returns the name of the object.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This example illustrates how to check if the Symbol text face is installed on the current machine.

```
// Example of how to check to see if a specific text face (Symbol)
// is installed on the current machine

fontName = "Symbol";
foundTextFace = false;

alert("No. typefaces: " + textFaces.length);

for (i=0; i<textFaces.length && foundTextFace == false; i++)
{
   fontToTest = textFaces[i];
   if (fontToTest.name == fontName)
   {
      foundTextFace = true;
   }
}

if (foundTextFace)
{
   alert(fontName + " is installed on this machine.");
}
else
{
   alert(fontName + " is not installed on this machine");
}
```

TextLine

A line of text in a specific text art object.

Properties

Property	R/O	Value type	What it is
autoKerning		Boolean	Should a font's built-in kerning information be used?
baseline		Number	Baseline of text.
baselineShift		Number	Baseline offset of text.
characters	R/O	Characters collection object	The characters contained in this text line.
clipping	R/O	Boolean	Is there a clipping path associated with the text art item containing this text line?
contents		String	The text contained in the text range.
direction		CharacterDirection constant	The orientation of the characters in a vertical text block.
evenodd		Boolean	Should the even-odd rule be used to determine insideness?
fillColor		Color	Fill color of text
filled		Boolean	Should the text be filled?
fillOverprint		Boolean	Should the art beneath the text be overprinted?
font		String	The text face of the text.
leading		Number	The vertical leading of the text.
length	R/O	Number	The number of character in the text.
note	R/O	String	The note associated with this text.
offset	R/O	Number	Offset of selected text in text range (in characters).
orientation	R/O	TextOrientation constant	The orientation of the text. Use the TextPath class to alter this property.
paragraph	R/O	Paragraph object	The paragraph containing this line of text.
parent	R/O	TextArtItem object	The parent of this object.
resolution	R/O	Number	The resolution of the object (in dots per inch).

Property	R/O	Value type	What it is
scaling		Array (of 2 numbers)	The character scaling supplied as a point with the first coordinate as horizontal scale and the second coordinate as vertical scale, where 100.0 is 100%.
size		Number	Font size of text.
strokeCap		StrokeCap constant	The type of line capping.
strokeColor		Color object	The stroke color for the path.
stroked		Boolean	Should the path be stroked?
strokeDashes		Array	Dash lengths. Set to an empty array for a solid line.
strokeDashOffset		Number	The default distance into the dash pattern at which the pattern should be started.
strokeJoin		StrokeJoin constant	Type of joints for the path.
strokeMiterLimit		Number	Are joins mitered (pointed) or beveled (squared-off)?
strokeOverprint		Boolean	Will art beneath a stroked object be overprinted?
strokeWidth		Number	Width of stroke.
stroked		Boolean	Is the TextLine stroked?
textPath	R/O	TextPath object	A reference to the text path associated with the text art item containing this text.
tracking		Number	The spacing between multiple characters.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
remove()	none	Nothing	Removes the referenced item from the document.

Method		Returns	What it does
textRange([rangeStart] [,rangeEnd])	 number number	TextRange object	Returns a TextRange object referencing a substring of the current text range, where rangeStart is the beginning character position and rangeEnd is the ending position. The first character position is zero. If omitted, rangeStart defaults to 0. If omitted, rangeEnd defaults to the last character of the range.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

`TextLine` gives you complete access to the text contained in a line of a text art object.

Lines of text cannot be created. When the `contents` property of a text art item is modified, Illustrator creates text lines as it reflows the text within the text art item.

Example

This example illustrates how to color lines of more than 80 characters red.

```
// Example of how to color lines of more than
// 80 characters red

// Make a reference to a red color

if (documents.length > 0)
{
   redRGB = new RGBColor();
   redColor = new Color();

    redRGB.red = 255;
    redRGB.green = 0;
    redRGB.blue = 0;
    redColor.rgb = redRGB;

   // Apply the red color to lines longer than 80 characters
   numTextArtItems = activeDocument.textArtItems.length;

   for (i=0; i < numTextArtItems; i++)
   {
      textArt = activeDocument.textArtItems[i];
      textArtRange = textArt.textRange();

      numLines = textArtRange.textLines.length;
      for (j=0; j < numLines; j++)
      {
         lineToExamine = textArtRange.textLines[j];
         if (lineToExamine.contents.length > 80)
         {
            lineToExamine.filled = true;
            lineToExamine.fillColor = redColor;
         }
      }
   }
}
```

TextLines

A collection of lines of text.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
removeAll	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Lines of text cannot be created. When the contents of a text art item is modified, Illustrator will create text lines as it reflows the text within the text art item.

Example

This script displays the total number of lines of text contained in all of the text art items in the current document.

```
// This script counts all text lines in current
// document and returns the total

if (documents.length > 0)
{
   numLines = 0;
   numTextArtItems = activeDocument.textArtItems.length;

   for (i=0; i < numTextArtItems; i++)
   {
      theText = activeDocument.textArtItems[i];
      textArtRange = theText.textRange();
      numLines = numLines + textArtRange.textLines.length;
   }

   alert("There are " + numLines + " lines of text in the document.");
}
```

TextPath

A text path. A text art item always has at least one text path.

Properties

Property	R/O	Value type	What it is
matrix		Matrix object	The transformation matrix for the text path.
name		String	The text path's name.
orientation		Orientation constant	The orientation of the text.
parent	R/O	TextArtItem object	The text art item that contains this text path.
textPathObject	R/O	PathItem object	Path associated with the text path (only valid for path text and area text).
textPathOffset		Number	The offset position where characters are anchored on the text path (only valid for path text).
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
remove()	none	Nothing	Removes the referenced item from the document.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Text paths provide access to a number of special properties for text art items. See Chapter 3 for additional information on text paths.

Example

This example illustrates how to set all text paths in the frontmost document to vertical.

```
// Example of how to change the orientation of all textpaths to vertical

if (documents.length > 0)
{
    for (i=0; i<activeDocument.textArtItems.length; i++)
   {
      textArt = activeDocument.textArtItems[i];
      for (j=0; j<textArt.textPaths.length; j++)
      {
         textArtPath = textArt.textPaths[j];
         textArtPath.orientation = TextOrientation.VERTICAL;
       }
    }
}
```

TextPaths

A collection of text paths in a specific text art item.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
add()	none	TextPath object	Creates a new object.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

See example under TextPath for a script that shows how to use the TextPaths collection.

TextPath_PathItems

A collection of path items associated with area text and path text.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This example illustrates how to create new on-path text. On-path text uses the first path specified in the TextPath_PathItems to shape the text.

```
// example of how to use the TextPath_PathItems collection
// to create an on-path text item

if (documents.length > 0)
{
    newTextArt = activeDocument.textArtItems.add();
    newTextArt.position = Array(200, 200);
    newTextArt.contents = "My new on-path text art";
    newTextArt.kind = TextType.PATHTEXT;

    newTextPath = newTextArt.textPath_PathItems[0];
    newTextPath.setEntirePath (Array(Array(200, 200), Array(250, 250),
Array(300, 200)));
}
```

TextRange

A range of text in a specific text art object.

Properties

Property	R/O	Value type	What it is
autoKerning		Boolean	Should a font's built-in kerning information be used?
baselineShift		Number	Baseline offset of text.
characters	R/O	Characters collection object	The characters contained in this text range.
clipping	R/O	Boolean	Is there a clipping path associated with the text art item containing this text range?
contents		String	The text contained in the text range.
direction		CharacterDirection constant	The orientation of the characters in a vertical text block.
evenodd		Boolean	Should the even-odd rule be used to determine insideness?
fillColor		Color	Fill color of text
filled		Boolean	Should the text be filled?
fillOverprint		Boolean	Should the art beneath the text be overprinted?
font		String	The text face of the text.
leading		Number	The vertical leading of the text.
length		Number	The number of character in the text.
note	R/O	String	The note associated with this text.
offset	R/O	Number	Offset of selected text in text range (in characters).
orientation	R/O	Orientation constant	The orientation of the text. Use the TextPath class to alter this property.
paragraphs	R/O	Paragraphs collection object	The paragraphs contained in this text range.
parent	R/O	TextArtItem object	The parent of this object.
resolution	R/O	Number	The resolution of the object (in dots per inch).

Property	R/O	Value type	What it is
scaling		Array (of 2 numbers)	The character scaling supplied as a point with the first coordinate as horizontal scale and the second coordinate as vertical scale, where 100.0 is 100%.
size		Number	Font size of text.
strokeCap		StrokeCap constant	The type of line capping.
strokeColor		color object	The stroke color for the path.
stroked		Boolean	Should the path be stroked?
strokeDashes		Array	Dash lengths. Set to an empty array for a solid line.
strokeDashOffset		Number	The default distance into the dash pattern at which the pattern should be started.
strokeJoin		StrokeJoin constant	Type of joints for the path.
strokeMiterLimit		Number	Are joins mitered (pointed) or beveled (squared-off)?
strokeOverprint		Boolean	Will art beneath a stroked object be overprinted?
strokeWidth		Number	Width of stroke.
textLines	R/O	TextLines collection object	The lines of text contained in this text range.
textPath	R/O	TextPath object	A reference to the text path associated with the text art item containing this text.
tracking		Number	The spacing between multiple characters.
typename	R/O	String	Returns the name of the referenced object.
words	R/O	Words collection object	The words contained in this text range.

Methods

Method		Returns	What it does
deleteRange()	none	Nothing	Deletes the text range.
remove()	none	Nothing	Removes the referenced item from the document.

Method		**Returns**	**What it does**
textRange([rangeStart] [,rangeEnd])	 number number	TextRange object	Returns a text range object referencing a substring of the current text range, where rangeStart is the beginning character position and rangeEnd is the ending position. The first character position is zero. If omitted, rangeStart defaults to 0. If omitted, rangeEnd defaults to the last character of the range.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

`TextRange` gives you access to the text contained in text art objects.

Example

This example illustrates how to resize the first part of every word in the frontmost document. The example illustrates how to obtain a sub-range from a text object.

```
// Example of how to use ranges and sub ranges to change the size of
// the first two characters of each word

if (documents.length > 0)
{
   for (i=0; i<activeDocument.textArtItems.length; i++)
   {
      textArt = activeDocument.textArtItems[i];
      textArtRange = textArt.textRange();

      for (j = 0 ; j < textArtRange.words.length; j++)
      {
         textWord = textArtRange.words[j];

         // For each word we check to see if it is longer
         // than 2 characters if it is we’ll resize the first
         // 2 characters if it is no we’ll resize the whole word

         wordLen = textWord.contents.length;

         if (wordLen < 2)
         {
            charsToChange = wordLen;
         }
         else
         {
            charsToChange = 2;
         }

         if (charsToChange > 0)
         {
            // Here we are obtaining a sub range. By leaving
            // the first argument out, we say: From the
            // beginning to character number charsToChange.
            // Note the first character in a TextRange has
            // an index of 0. We therefore have to subtract 1

            firstChars = textWord.textRange(0, charsToChange - 1 );
            firstChars.size = firstChars.size * 1.5;
         }
      }
   }
}
```

Variable

A class of document-level variables that can be imported or exported.

Properties

Property	R/O	Value type	What it is
kind		VariableKind constant	The variable’s type.
name		String	The name of this variable.
pageItems		PageItems object	The collection of pageItems in this document.
parent		Object	The object that contains the variable.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
remove()	none	Nothing	Removes the variable from the collection of variables.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Variables

The collection of variables in the document.

Properties

Property	R/O	Value type	What it is
length		Number	The number of variables in the document.
parent		Object	The object that contains the collection of variables.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	Variable object	Adds a variableObject to the collection of variables
removeAll()	none	Nothing	Removes all the variables from the collection of variables.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

View

A document view in an Illustrator document.

Properties

Property	R/O	Value type	What it is
bounds	R/O	Array (of 4 numbers)	The bounding rectangle of this view relative to the current document's bounds.
centerPoint		Array (of 2 numbers)	The center point of this view relative to the current document's bounds.
parent	R/O	Document object	The document that contains this view.
screenMode		ScreenMode constant	The mode of display for this view.
typename	R/O	String	Returns the name of the referenced object.
zoom		Number	The zoom factor of this view, where 100.0 is 100%.

Methods

Method	Parameter type	Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

Illustrator's `view` object represents a window view onto a document. New views cannot be created, but some properties of existing views can be modified, including the center point, screen mode and zoom.

Example

This example illustrates how to set the first view of the frontmost document to full screen mode.

```
// Example of how to set the first view of the frontmost document
// to full screen

if (documents.length > 0)
{
   documents[0].views[0].screenMode = ScreenMode.FULLSCREEN;
}
```

Views

A collection of views in a document.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

See the example under View for a sample script that shows how to use the Views collection.

Word

A string of text in a textArtItem that is separated by whitespace.

Properties

Property	R/O	Value type	What it is
autoKerning		Boolean	Should a font’s built-in kerning information be used?
baselineShift		Number	Baseline offset of text.
characters	R/O	characters collection object	The characters contained in this word.
clipping	R/O	Boolean	Is there a clipping path associated with the text art item containing this word?
contents		String	The text contained in the text range.
direction		CharacterDirection constant	The orientation of the characters in a vertical text block.
evenodd		Boolean	Should the even-odd rule be used to determine insideness?
fillColor		Color	Fill color of text
filled		Boolean	Should the text be filled?
fillOverprint		Boolean	Should the art beneath the text be overprinted?
font		String	The text face of the text.
leading		Number	The vertical leading of the text.
length	R/O	Number	The number of character in the text.
note	R/O	String	The note associated with this text.
offset	R/O	Number	Offset of selected text in text range (in characters).
orientation	R/O	TextOrientation constant	The orientation of the text. Use the TextPath class to alter this property.
paragraph	R/O	Paragraph object	The paragraph containing the character.
parent	R/O	TextArtItem object	The parent of this object.
resolution	R/O	Number	The resolution of the object (in dots per inch).

Property	R/O	Value type	What it is
scaling		Array (of 2 numbers)	The character scaling supplied as a point with the first coordinate as horizontal scale and the second coordinate as vertical scale, where 100.0 is 100%.
size		Number	Font size of text.
strokeCap		StrokeCap constant	The type of line capping.
strokeColor		color object	The stroke color for the path.
stroked		Boolean	Should the path be stroked?
strokeDashes		Array	Dash lengths. Set to an empty array for a solid line.
strokeDashOffset		Number	The default distance into the dash pattern at which the pattern should be started.
strokeJoin		StrokeJoin constant	Type of joints for the path.
strokeMiterLimit		Number	Are joins mitered (pointed) or beveled (squared-off)?
strokeOverprint		Boolean	Will art beneath a stroked object be overprinted?
strokeWidth		Number	Width of stroke.
textPath	R/O	TextPath object	A reference to the text path associated with the text art item containing this text.
tracking		Number	The spacing between multiple characters.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method		Returns	What it does
remove()	none	Nothing	Removes the referenced item from the document.

Method		**Returns**	**What it does**
textRange([rangeStart] [,rangeEnd])	 number number	TextRange object	Returns a text range object referencing a substring of the current text range, where rangeStart is the beginning character position and rangeEnd is the ending position. The first character position is zero. If omitted, rangeStart defaults to 0. If omitted, rangeEnd defaults to the last character of the range.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Notes

`Word` gives you complete access to the individual words contained in text art objects in Illustrator.

Example

This example illustrates how to color every instance of the word “the.”

```
// Example of how to color certain words

// Create the color to apply to the words

if (documents.length > 0 && activeDocument.textArtItems.length > 0)
{
   wordColor = new RGBColor();
   newColor = new Color();

    wordColor.red = 255;
    wordColor.green = 0;
    wordColor.blue = 255;
    newColor.rgb = wordColor;

   // Set the value of the word to look for

    searchWord = "the";

   // Iterate through all words in the document
   // and color the words that macth searchWord

    for (i=0; i<activeDocument.textArtItems.length; i++)
   {
       textArt = activeDocument.textArtItems[i];

       textArtRange = textArt.textRange();

       for (j=0; j<textArtRange.words.length; j++)
      {
           theWord = textArtRange.words[j];

           if (theWord.contents == searchWord)
         {
               theWord.filled = true;
               theWord.fillColor = newColor;
           }
       }
    }
}
else
{
   alert("there are no text art items");
}
```

Words

A collection of words.

Properties

Property	R/O	Value type	What it is
length	R/O	Number	The number of objects in the collection.
parent	R/O	Object	The parent of this object.
typename	R/O	String	Returns the name of the referenced object.

Methods

Method	Parameter type	Returns	What it does
add()	none	Word object	Add a word to the contents of a text art object.
addBefore()	none	Nothing	Adds a word before the current word selection or insertion point.
removeAll()	none	Nothing	Deletes all objects in this collection.
toString()	none	String	Returns the object type of a referenced object. If the object has a name, also returns the name.

Example

This script displays the total number of words contained in all of the text art items in the current document.

```
// This script counts all words in current document and reports the total

if (documents.length > 0)
{
   numWords = 0;
   for (i=0; i<activeDocument.textArtItems.length; i++)
   {
      theTextArt = activeDocument.textArtItems[i];
      textArtRange = theTextArt.textRange();
      numWords = numWords + textArtRange.words.length;
   }
   alert("There are " + numWords + " words in the document.");
}
```

2 Scripting Constants

Constant Type	Values	What it means
BlendModes	COLORBLEND	The blend mode used when compositing an object.
	COLORBURN	
	COLORDODGE	
	DARKEN	
	DIFFERENCE	
	EXCLUSION	
	HARDLIGHT	
	HUE	
	LIGHTEN	
	LUMINOSITY	
	MULTIPLY	
	NORMAL	
	OVERLAY	
	SATURATIONBLEND	
	SCREEN	
	SOFTLIGHT	
CharacterDirection	KUMIMOJI	The orientation of the characters in a vertical text block.
	NORMAL	
	ROTATED	

Constant Type	Values	What it means
ColorDitherMethod	DIFFUSION	The method used to dither colors in exported GIF and PNG8 images.
	NOISE	
	NOREDUCTION	
	PATTERNDITHER	
ColorModel	PROCESS	
	REGISTRATION	
	SPOT	
ColorReductionMethod	ADAPTIVE	The method used to reduce the number of colors in exported GIF and PNG8 images.
	PERCEPTUAL	
	SELECTIVE	
	WEB	
ColorType	CMYK	The color specification for an individual color.
	GRADIENT	
	GRAY	
	NONE	
	PATTERN	
	RGB	
	SPOT	
Compatibility	ILLUSTRATOR10	The version of the Illustrator file to create when saving an EPS or Illustrator file.
	ILLUSTRATOR3	
	ILLUSTRATOR4	
	ILLUSTRATOR5	
	ILLUSTRATOR6	
	ILLUSTRATOR7	
	ILLUSTRATOR8	
	ILLUSTRATOR9	

Constant Type	**Values**	**What it means**
CompressionQuality	AUTOMATIC	The quality of bitmap compression used when saving a PDF file.
	JPEGHIGH	
	JPEGLOW	
	JPEGMAXIMUM	
	JPEGMEDIUM	
	JPEGMINIMUM	
	NONE	
	ZIP4BIT	
	ZIP8BIT	
Crop Options	JAPANESE	The style of a document’s cropping box.
	STANDARD	
DocumentColorSpace	CMYK	The color space of a document.
	RGB	
DocumentType	EPS	The file format used to save a file.
	ILLUSTRATOR	
	PDF	
EPSPreview	BWMACINTOSH	The preview image format used when saving an EPS file.
	BWTIFF	
	COLORMACINTOSH	
	COLORTIFF	
	NONE	
	TRANSPARENTCOLORTIFF	

Constant Type	Values	What it means
ExportType	FLASH	The file format used to export a file.
	GIF	
	JPEG	
	PHOTOSHOP	
	PNG24	
	PNG8	
	SVG	
FlashExportStyle	ASFLASHFILE	The method used to convert Illustrator images when exporting files.
	LAYERSASFILES	
	LAYERSASFRAMES	
FlashImageFormat	LOSSLESS	The format used to store flash images.
	LOSSY	
FlashJPEGMethod	OPTIMIZED	The method used to store JPEG images.
	STANDARD	
GradientType	LINEAR	The type of gradient.
	RADIAL	
ImageColorSpace	CMYK	The color space of a raster item or an exported Photoshop 5 file.
	GRAYSCALE	
	RGB	
Justification	ALLLINES	The alignment or justificaton for a paragraph of text.
	CENTER	
	FULLLINES	
	LEFT	
	RIGHT	
	UNKNOWN	

Constant Type	Values	What it means
KnockoutState	DISABLED	The type of knockout to use on a page item.
	ENABLED	
	INHERITED	
	UNKNOWN	
MonochromeCompression	CCIT3	The type of compression to use on a monochrome bitmap item when saving a PDF file.
	CCIT4	
	MONOZIP	
	NONE	
	RUNLENGTH	
OutputFlattening	PRESERVEAPPEARANCE	How transparency should be flattened when saving EPS and Illustrator file formats with compatibility set to versions of Illustrator earlier than Illustrator 10.
	PRESERVEPATHS	
PathPointSelection	ANCHORPOINT	Which points, if any, of a path are selected.
	LEFTDIRECTION	
	LEFTRIGHTPOINT	
	NOSELECTION	
	RIGHTDIRECTION	
PDFCompatibility	ACROBAT4	The version of the Acrobat file format to create when saving a PDF file.
	ACROBAT5	
PointType	CORNER	The type of path point selected.
	SMOOTH	
PolarityValues	NEGATIVE	
	POSITIVE	
PostScriptLevel	LEVEL1	The PostScript level to use when saving and EPS file.
	LEVEL2	
	LEVEL3	

Constant Type	Values	What it means
RasterLinkState	DATAFROMFILE	The status of a raster item's linked image if the image is stored externally.
	DATAMODIFIED	
	NODATA	
RulerUnits	CENTIMETERS	The default measurement units for the rulers of a document.
	INCHES	
	MILLIMETERS	
	PICAS	
	POINTS	
	QS	
	UNKNOWN	
SaveOptions	DONOTSAVECHANGES	Save options provided when closing a document.
	PROMPTTOSAVECHANGES	
	SAVECHANGES	
ScreenMode	DESKTOP	The mode of display for a view.
	FULLSCREEN	
	MULTIWINDOW	
StrokeCap	BUTTENDCAP	The type of line capping for a path stroke.
	PROJECTINGENDCAP	
	ROUNDENDCAP	
StrokeJoin	BEVELENDJOIN	The type of joints for a path stroke.
	MITERENDJOIN	
	ROUNDENDJOIN	
SVGCSSPropertyLocation	ENTITIES	How should the CSS properties of the document be included in an exported SVG file.
	PRESENTATIONATTRIBUTES	
	STYLEATTRIBUTES	
	STYLEELEMENTS	

Constant Type	Values	What it means
SVGDocumentEncoding	ASCII	How should the text in the document be encoded when exporting an SVG file.
	UTF16	
	UTF8	
SVGFontSubsetting	ALLGLYPHS	What font glyphs should be included in the exported SVG file.
	COMMONENGLISH	
	COMMONROMAN	
	GLYPHSUSED	
	GLYPHSUSEDPLUSENGLISH	
	GLYPHSUSEDPLUSROMAN	
	NONE	
TabStopAlignment	CENTER	The alignment of a tab stop.
	DECIMAL	
	LEFT	
	RIGHT	
	UNKNOWN	
TextOrientation	HORIZONTAL	The orientation of text in a textArt item.
	VERTICAL	
TextType	AREATEXT	The type of textArt displayed by this object.
	PATHTEXT	
	POINTTEXT	

Constant Type	Values	What it means
Transformation	BOTTOM	The point to use as the anchor point about which an object is rotated, resized, or transformed.
	BOTTOMLEFT	
	BOTTOMRIGHT	
	CENTER	
	DOCUMENTORIGIN	
	LEFT	
	RIGHT	
	TOP	
	TOPLEFT	
	TOPRIGHT	
UserInteractionLevel	DISPLAYALERTS	User interface settings.
	DONTDISPLAYALERTS	
VariableKind	GRAPH	What type of variables are included in the document.
	IMAGE	
	TEXTUAL	
	UNKNOWN	
	VISIBILITY	
ZOrderMethod	BRINGFORWARD	The method used to arrange an art object's position in the stacking order of its parent group or layer, as specified with the zOrder method.
	BRINGTOFRONT	
	SENDBACKWARD	
	SENDTOBACK	

Index