ADOBE® ILLUSTRATOR® CC 2015

ADOBE ILLUSTRATOR CC 2015.3 SCRIPTING REFERENCE: JAVASCRIPT

Adobe Illustrator CC 2015 Scripting Reference: JavaScript

Contents

1 JavaScript Object Reference

This section presents all of the object classes in the 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.
- Sample code to help illustrate the syntax and typical workflow usage of the object class.

  These examples are intended to be clear demonstrations of syntax, and do not show the best or most efficient way to construct a JavaScript script. Error checking, for instance, is generally brief or missing. However, the examples can be combined and expanded to make scripts with greater functionality.

Application

The Adobe® Illustrator® application object, referenced using the pre-defined global `app` object, which contains all other Illustrator objects.

Application properties

Property	Value type	What it is
`activeDocument`	`Document`	The active (frontmost) document in Illustrator.
`browserAvailable`	`boolean`	Read-only. If `true`, a web browser is available.
`buildNumber`	`string`	Read-only. The application's build number.
`colorSettingsList`	object	Read-only. The list of color-settings files currently available for use.
`coordinateSystem`	`CoordinateSystem`	The coordinate system currently in use, document or artboard.
`defaultColorSettings`	`File`	Read-only. The default color-settings file for the current application locale.
`documents`	`Documents`	Read-only. The documents in the application.
`flattenerPresetList`	object	Read-only. The list of flattener style names currently available for use.
`freeMemory`	`number (long)`	Read-only. The amount of unused memory (in bytes) within the Illustrator partition.
`locale`	`string`	Read-only. The application's locale.
`name`	`string`	Read-only. The application's name (not related to the filename of the application file).
`pasteRememberLayers`	`boolean`	Read-only. If `true`, the paste operation maintains the layer structure.
`path`	`File`	Read-only. The file path to the application.
`PDFPresetsList`	object	Read-only. The list of preset PDF-options names available for use.
`PPDFileList`	object	Read-only. The list of PPD files currently available for use.
`preferences`	`Preferences`	Illustrator's preference settings.
`printerList`	`array` of `Printer`	Read-only. The list of installed printers.
`printPresetsList`	object	Read-only. The list of preset printing-options names available for use.
`scriptingVersion`	`string`	Read-only. The version of the Scripting plug-in.

Property	Value type	What it is
selection	array of objects	All currently selected objects in the active (frontmost) document.
startupPresetsList	object	Read-only. The list of presets available for creating a new document.
textFonts	TextFonts	Read-only. The installed fonts.
tracingPresetList	array of string	Read-only. The list of preset tracing-options names available for use.
typename	string	Read-only. The class name of the referenced object.
userInteractionLevel	UserInteractionLevel	What level of interaction with the user should be allowed when handling script commands.
version	string	Read-only. The application's version.
visible	boolean	Read-only. If true, the application is visible.

Application methods

Method	Parameter type	Returns	What it does
beep ()		nothing	Alerts the user.
concatenateMatrix (matrix, secondMatrix)	 Matrix Matrix	Matrix	Joins two matrices together.
concatenateRotationMatrix (matrix, angle)	 Matrix number (double)	Matrix	Joins a rotation translation to a transformation matrix.
concatenateScaleMatrix (matrix [,scaleX] [,scaleY])	 Matrix number (double) number (double)	Matrix	Concatenates a scale translation to a transformation matrix.
concatenateTranslationMatrix (matrix [,deltaX] [,deltaY])	 Matrix number (double) number (double)	Matrix	Joins a translation to a transformation matrix.
convertSampleColor (sourceColorSpace, sourceColor, destColorSpace, colorConvertPurpose [,sourceHasAlpha] [,destHasAlpha])	 ImageColorSpace ColorComponents ImageColorSpace ColorConvertPurpose boolean boolean	array of ColorComponents	Converts a sample-component color from one color space to another.
copy ()		nothing	Copies current selection to the clipboard.

Method	Parameter type	Returns	What it does
`cut ()`		nothing	Cuts current selection to the clipboard.
`deleteWorkspace (workspaceName)`	`string`	Boolean	Deletes an existing workspace.
`getIdentityMatrix ()`		`Matrix`	Returns an identity matrix.
`getPPDFileInfo (name)`	`string`	`PPDFileInfo`	Gets detailed file information for specified PPD file.
`getPresetFileOfType (presetType)`	`DocumentPresetType`	`File`	Returns the full path to the application's default document profile for the specified preset type.
`getPresetSettings (preset)`	`string`	`DocumentPreset`	Retrieves the tracing-option settings from the template with a given preset name.
`getRotationMatrix ([angle])`	`number (double)`	`Matrix`	Returns a transformation matrix containing a single rotation. **NOTE:** Requires a value in degrees. For example, 30 rotates the object 30 degrees counterclockwise; -30 rotates the object 30 degrees clockwise.
`getScaleMatrix ([scaleX] [, scaleY])`	`number (double)` `number (double)`	`Matrix`	Returns a transformation matrix containing a single scale. **NOTE:** Requires a value in percentage. For example, 60 scales the object to 60% of its original size; 200 doubles the object's bounds.
`getScriptableHelpGroup ()`		`variant`	Gets the scriptable help group object that represents the search widget in the app bar.
`getTranslationMatrix ([deltaX] [, deltaY])`	`number (double)` `number (double)`	`Matrix`	Returns a transformation matrix containing a single translation. **NOTE:** Requires a value in points. For example, ({100,200} moves the object 100 pt. to the right and 200 pt. up; a minus before each number moves the object left and down.
`invertMatrix (matrix)`	`Matrix`	`Matrix`	Inverts a matrix.

Method	Parameter type	Returns	What it does
`isEqualMatrix (matrix, secondMatrix)`	`Matrix` `Matrix`	`boolean`	Checks whether the two matrices are equal.
`isSingularMatrix (Matrix)`	`Matrix`	`boolean`	Checks whether a matrix is singular and cannot be inverted.
`loadColorSettings (fileSpec)`	`File`	nothing	Loads color settings from specified file, or, if file is empty, turns color management off.
`open (file [, documentColorSpace] [, options])`	`File` `DocumentColorSpace` anything	`Document`	Opens the specified document file. If you open a pre-Illustrator 9 document that contains both RGB and CMYK colors and `documentColorSpace` is supplied, all colors are converted to the specified color space. If the parameter is not supplied, Illustrator opens a dialog so the user can choose the color space.
`paste()`		nothing	Pastes current clipboard content into the current document.
`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.
`redo()`		nothing	Redoes the most recently undone transaction.
`redraw ()`		nothing	Forces Illustrator to redraw all its windows.
`resetWorkspace() ()`		Boolean	Resets the current workspace.
`saveWorkspace() (workspaceName)`	`string`	Boolean	Saves a new workspace.
`sendScriptMessage (pluginName, messageSelector, inputString)`	`string` `string` `string`	`string`	Sends a plug-in-defined command message to a plug-in with given input arguments, and returns the plug-in-defined result string.
`showPresets (fileSpec)`	`File`	`PrintPresetList`	Gets presets from the file.
`switchWorkspace() (workspaceName)`	`string`	Boolean	Switches to the specified workspace.

Method	Parameter type	Returns	What it does
`translatePlaceholderText (text)`	`string`	`string`	Translates the placeholder text to regular text (a way to enter Unicode points in hex values).
`undo()`		nothing	Undoes the most recent transaction.

Duplicating the active document

```
// Duplicates any selected items from
// the active document into a new document.

var newItem;
var docSelected = app.activeDocument.selection;

if ( docSelected.length > 0 ) {
    // Create a new document and move the selected items to it.
    var newDoc = app.documents.add();
    if ( docSelected.length > 0 ) {
        for ( i = 0; i < docSelected.length; i++ ) {
            docSelected[i].selected = false;
            newItem = docSelected[i].duplicate( newDoc,
                   ElementPlacement.PLACEATEND );
        }
    }
    else {
        docSelected.selected = false;
        newItem = docSelected.parent.duplicate( newDoc,
               ElementPlacement.PLACEATEND );
    }
}
else {
    alert( "Please select one or more art objects" );
}
```

Artboard

An `Artboard` object represents a single artboard in a document. There can be between 1 to 100 artboards in one document.

Artboard properties

Property	Value type	What it is
`artboardRect`	`rect`	Size and position of the artboard.
`name`	`string`	The unique identifying name of the artboard.
`parent`	`Document`	Read-only. The parent of this object.
`rulerOrigin`	`Point`	Ruler origin of the artboard, relative to the top left corner of the artboard.
`rulerPAR`	`number (double)`	Pixel aspect ratio, used in ruler visualization if the units are pixels. Range: 0.1 to 10.0
`showCenter`	`boolean`	Show center mark.
`showCrossHairs`	`boolean`	Show cross hairs.
`showSafeAreas`	`boolean`	Show title and action safe areas (for video).
`typename`	`string`	Read-only. The class name of this object.

Artboards methods

Method	Parameter type	Returns	What it does
`remove ()`		Nothing	Deletes this artboard object. You cannot remove the last artboard in a document.

Artboards

A collection of `Artboard` objects.

Artboards properties

Property	Value type	What is it
`length`	`number`	Read-only. The number of datasets in the collection
`parent`	`Artboard`	Read-only. The name of the object that contains this dataset
`typename`	`string`	Read-only. The class name of the referenced object.

Artboards methods

Method	Parameter type	Returns	What it does
`add (artboardRect)`	`rect`	`Artboard`	Creates a new `Artboard` object.
`getActiveArtboardIndex ()`		`number (long)`	Retrieves the index position of the active artboard in the document's list. Returns the 0-based index.
`getByName (name)`	`string`	`Artboard`	Gets the first element in the collection with the specified name.
`insert (artboardRect, index)`	`rect` `number (long)`	Nothing	Creates a new `Artboard` object and inserts it at the given index in the list.
`remove (index)`	`number (long)`	Nothing	Deletes an artboard object. You cannot remove the last artboard in a document.
`setActiveArtboardIndex (index)`	`number (long)`	Nothing	Makes a specific artboard active and makes it current in the iteration order.

Brush

A brush in an Illustrator document. Brushes are contained in documents. Additional brushes may be created by the user within Illustrator. You can access brushes within a script, but you cannot create them.

Brush properties

Property	Value type	What it is
`name`	`string`	The name of the `brush`.
`parent`	`Document`	Read-only. The document that contains this `brush`.
`typename`	`string`	Read-only. The class name of the referenced object.

Brush methods

Method	Parameter type	Returns	What it does
`applyTo (artItem)`	`PageItem`	Nothing	Applies the `brush` to a specific art item.

Applying a brush

```
// Duplicates and groups all items in the current selection,
// then applies the same brush to each item in the group

if ( app.documents.length > 0 ) {
    docSelection = app.activeDocument.selection;
    if ( docSelection.length > 0 ) {
        newGroup = app.activeDocument.groupItems.add();

        for ( i = 0; i < docSelection.length; i++ ) {
            newItem = docSelection[i].duplicate();
            newItem.moveToBeginning( newGroup );
        }
        brush4 = app.activeDocument.brushes[1];
        brush4.applyTo( newGroup );
    }
}
```

Brushes

A collection of `brush` objects in a document.

Brushes properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The document that contains this brushes collection.
`typename`	`string`	Read-only. The class name of the referenced object.

Brushes methods

Method	Parameter type	Returns	What it does
`getByName (name)`	`string`	`Brush`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`Brush`	Gets an element from the collection.

Counting brushes

```
// Counts all brushes in the active document

if ( app.documents.length > 0 ) {
    numberOfBrushes = app.activeDocument.brushes.length;
}
```

CharacterAttributes

Specifies the properties of a character contained in a text frame. A `CharacterStyle` object associates these attributes with a specific text range through its `characterAttributes` property.

NOTE: Character attributes do not have default values, and are undefined until explicitly set.

CharacterAttributes properties

Property	Value type	What it is
`akiLeft`	`number (double)`	The amount of inter-character spacing to be added to the left side of the character, in thousandths of an em (that amount will not compress or expand during full-justification).
`akiRight`	`number (double)`	The amount of inter-character spacing to be added to the right side of the character, in thousandths of an em (that amount will not compress or expand during full-justification).
`alignment`	`StyleRunAlignmentType`	The character alignment type.
`alternateGlyphs`	`AlternateGlyphsForm`	The alternate glyphs form.
`autoLeading`	`boolean`	If `true`, the automatic leading should be used.
`baselineDirection`	`BaselineDirectionType`	The Japanese text baseline direction.
`baselinePosition`	`FontBaselineOption`	The baseline position of text.
`baselineShift`	`number (double)`	The amount of shift in points of the text baseline.
`capitalization`	`FontCapsOption`	The case of text.
`connectionForms`	`boolean`	If `true`, the OpenType® connection forms should be used.
`contextualLigature`	`boolean`	If `true`, the contextual ligature should be used.
`discretionaryLigature`	`boolean`	If `true`, the discretionary ligature should be used.
`figureStyle`	`FigureStyleType`	The number style in an OpenType font.
`fillColor`	`Color`	The color of the text fill.

Property	Value type	What it is
fractions	boolean	If true, the OpenType fractions should be used.
horizontalScale	number (double)	The character horizontal scaling factor expressed as a percentage (100 = 100%).
italics	boolean	If true, the Japanese OpenType font supports italics.
kerningMethod	AutoKernType	The automatic kerning method to use.
language	LanguageType	The language of text.
leading	number (double)	The amount of space between two lines of text, in points.
ligature	boolean	If true, the ligature should be used.
noBreak	boolean	If true, line breaks are not allowed.
openTypePosition	FontOpenTypePositionOption	The OpenType baseline position.
ordinals	boolean	If true, the OpenType ordinals should be used.
ornaments	boolean	If true, the OpenType ornaments should be used.
overprintFill	boolean	If true, the fill of the text should be overprinted.
overprintStroke	boolean	If true, the stroke of the text should be overprinted.
parent	object	Read-only. The object's container.
proportionalMetrics	boolean	If true, the Japanese OpenType font supports proportional glyphs.
rotation	number (double)	The character rotation angle in degrees.
size	number (double)	Font size in points.
strikeThrough	boolean	If true, characters use strike-through style.
strokeColor	Color	The color of the text stroke.
strokeWeight	number (double)	Line width of stroke.

Property	Value type	What it is
stylisticAlternates	boolean	If true, the OpenType stylistic alternates should be used.
swash	boolean	If true, the OpenType swash should be used.
tateChuYokoHorizontal	number (long)	The Tate-Chu-Yoko horizontal adjustment in points.
tateChuYokoVertical	number (long)	The Tate-Chu-Yoko vertical adjustment in points.
textFont	TextFont	The text font.
titling	boolean	If true, the OpenType titling alternates should be used.
tracking	number (long)	The tracking or range kerning amount, in thousandths of an em.
Tsume	number (double)	The percentage of space reduction around a Japanese character.
typename	string	Read-only. The class name of the object.
underline	boolean	If true, characters are underlined.
verticalScale	number (double)	Character vertical scaling factor expressed as a percentage (100 = 100%).
wariChuCharactersAfterBreak	number (long)	Specifies how the characters in Wari-Chu text (an inset note in Japanese text) are divided into two or more lines.
wariChuCharactersBeforeBreak	number (long)	Specifies how the characters in Wari-Chu text (an inset note in Japanese text) are divided into two or more lines.
wariChuEnabled	boolean	If true, Wari-Chu is enabled.
wariChuJustification	WariChuJustificationType	The Wari-Chu justification.
wariChuLineGap	number (long)	The Wari-Chu line gap.
wariChuLines	number (long)	The number of Wari-Chu (multiple text lines fit into a space meant for one) lines.
wariChuScale	number (double)	The Wari-Chu scale.

Setting character attributes

```
// Creates a new document, adds a simple text item
// then incrementally increases the horizontal and
// vertical scale attributes of each character

var docRef = documents.add();
var textRef = docRef.textFrames.add();
textRef.contents = "I Love Scripting!";
textRef.top = 400;
textRef.left = 100;

// incrementally increase the scale of each character
var charCount = textRef.textRange.characters.length;
var size = 100;
for(i=0; i<charCount; i++, size *= 1.2) {
    textRef.textRange.characters[i].characterAttributes.horizontalScale
        = size;
    textRef.textRange.characters[i].characterAttributes.verticalScale
        = size;
}
```

Characters

A collection of characters (`TextRange` objects of length 1). The elements are not named; you must access them by index.

Characters properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of characters in the collection.
`parent`	object	Read-only. The text art item that contains this character.
`typename`	`string`	Read-only. The class name of the referenced object.

Characters methods

Method	Parameter type	Returns	What it does
`add (contents [,relativeObject] [,insertionLocation])`	`string` `TextFrameItem` `ElementPlacement`	`TextRange`	Adds a new character with specified text contents at the specified location in the current document. If a location is not specified, adds the new character to the containing text frame after the current text selection or insertion point.
`addBefore (contents)`	`string`	`TextRange`	Adds a character before the specified text selection.
`index (itemKey)`	`number`	`TextRange`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in the collection.

Counting characters

```
// Counts all characters in the active document,
// including whitespace, and stores in numChars

if ( app.documents.length > 0 ) {
    var doc = app.activeDocument;
    var numChars = 0;
    for ( i = 0; i < doc.textFrames.length; i++ ) {
        textArtRange = doc.textFrames[i].contents;
        numChars += textArtRange.length;
    }
}
```

CharacterStyle

Associates character attributes with characters. For an example, see CharacterStyles.

CharacterStyle properties

Property	Value type	What it is
`characterAttributes`	`CharacterAttributes`	Read-only. The character properties for the style.
`name`	`string`	The character style's name.
`parent`	object	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the object.

CharacterStyle methods

Method	Parameter type	Returns	What it does
`applyTo (textItem [,clearingOverrides])`	object `boolean`	Nothing	Applies the character style to the text object or objects.
`remove ()`		Nothing	Deletes the object.

CharacterStyles

A collection of `CharacterStyle` objects.

CharacterStyles properties

Property	Value type	What it is
`length`	`number`	Read-only. Number of elements in the collection.
`parent`	object	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the object.

CharacterStyles methods

Method	Parameter type	Returns	What it does
`add (name)`	`string`	`CharacterStyle`	Creates a named character style.
`getByName (name)`	`string`	`CharacterStyle`	Gets the first element in the collection with the provided name.
`index (itemKey)`	`string, number`	`CharacterStyle`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Using characters styles

```
// Creates 3 text frames in a new document then creates
// a character style and applies it to each text frame.

var docRef = documents.add();
var textRef1 = docRef.textFrames.add();
textRef1.contents = "Scripting is fun!";
textRef1.top = 700;
textRef1.left = 50;

var textRef2  = docRef.textFrames.add();
textRef2.contents = "Scripting is easy!";
textRef2.top = 625;
textRef2.left = 100;

var textRef3  = docRef.textFrames.add();
textRef3.contents = "Everyone should script!";
textRef3.top = 550;
textRef3.left = 150;
redraw();

// Create a new character style
var charStyle = docRef.characterStyles.add("BigRed");
```

```
// set character attributes
var charAttr = charStyle.characterAttributes;
charAttr.size = 40;
charAttr.tracking = -50;
charAttr.capitalization = FontCapsOption.ALLCAPS;
var redColor = new RGBColor();
redColor.red = 255;
redColor.green = 0;
redColor.blue = 0;
charAttr.fillColor = redColor;

// apply to each textFrame in the document
charStyle.applyTo(textRef1.textRange);
charStyle.applyTo(textRef2.textRange);
charStyle.applyTo(textRef3.textRange);
```

CMYKColor

A CMYK color specification, used where a `color` object is required.

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.

CMYKColor properties

Property	Value type	What it is
`black`	`number (double)`	The black color value. Range 0.0–100.0. Default: 0.0
`cyan`	`number (double)`	The cyan color value. Range 0.0–100.0. Default: 0.0
`magenta`	`number (double)`	The magenta color value. Range 0.0–100.0. Default: 0.0
`typename`	`string`	Read-only. The class name of the referenced object.
`yellow`	`number (double)`	The yellow color value. Range 0.0–100.0. Default: 0.0

Setting a CMYK color

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

if ( app.documents.length > 0 &&  app.activeDocument.pathItems.length > 0) {
    frontPath = app.activeDocument.pathItems[0];
    // Set color values for the CMYK object
    newCMYKColor = new CMYKColor();
    newCMYKColor.black = 0;
    newCMYKColor.cyan = 30.4;
    newCMYKColor.magenta = 32;
    newCMYKColor.yellow = 0;
    // Use the color object in the path item
    frontPath.filled = true;
    frontPath.fillColor = newCMYKColor;
}
```

Color

An abstract parent class for all color classes used in Illustrator. Subclasses are:

CMYKColor
GradientColor
GrayColor
LabColor
NoColor
PatternColor
RGBColor
SpotColor

CompoundPathItem

A compound path. These objects are composed of multiple intersecting paths, resulting in transparent interior spaces where the component paths overlap. The `pathItems` property provides access to the paths that make up the compound path.

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

All paths within 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, the properties of all other component paths are updated with the new value.

CompoundPathItem properties

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout.
`blendingMode`	`BlendModes`	The mode used when compositing an object.
`controlBounds`	`array` of 4 numbers	Read-only. The bounds of the object including stroke width and controls.
`editable`	`boolean`	Read-only. If `true`, this item is editable.
`geometricBounds`	`array` of 4 numbers	Read-only. The bounds of the object excluding stroke width.
`height`	`number(double)`	The height of the compound path item excluding stroke width.
`hidden`	`boolean`	If `true`, this compound path item is hidden.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layer`	`Layer`	Read-only. The layer to which this compound path item belongs.
`left`	`number(double)`	The position of the left side of the item (in points, measured from the left side of the page).
`locked`	`boolean`	If `true`, this compound path item is locked.
`name`	`string`	The name of this compound path item.
`note`	`string`	The note assigned to this item.
`opacity`	`number (double)`	The opacity of the object. Range: 0.0 to 100.0
`parent`	`Layer` or `GroupItem`	Read-only. The parent of this object.
`pathItems`	`PathItems`	Read-only. The path art items in this compound path.

Property	Value type	What it is
position	array of 2 numbers	The position (in points) of the top left corner of the compoundPathItem object in the format [x, y]. Does not include stroke weight.
selected	boolean	If true, this compound path item is selected.
sliced	boolean	If true, the item is sliced. Default: false
tags	Tags	Read-only. The tags contained in this object.
top	number (double)	The position of the top of the item (in points, measured from the bottom of the page).
typename	string	Read-only. Read-only. The class name of the referenced object.
uRL	string	The value of the Adobe URL tag assigned to this compound path item.
visibilityVariable	Variant	The visibility variable bound to the item.
visibleBounds	array of 4 numbers	Read-only. The visible bounds of the compound path item including stroke width.
width	number (double)	The width of the compound path item excluding stroke width.
wrapInside	boolean	If true, the text frame object should be wrapped inside this object.
wrapOffset	number (double)	The offset to use when wrapping text around this object.
wrapped	boolean	If true, wrap text frame objects around this object (text frame must be above the object).
zOrderPosition	number (long)	Read-only. The position of this art item within the stacking order of the group or layer (Parent) that contains the art item.

CompoundPathItem methods

Method	Parameter type	Returns	What it does
`duplicate ([relativeObject] [,insertionLocation])`	object `ElementPlacement`	`CompoundPathItem`	Creates a duplicate of the selected object.
`move (relativeObject, insertionLocation)`	object `ElementPlacement`	Nothing	Moves the object.
`remove ()`		Nothing	Deletes this object.
`resize (scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])`	`number (double)` `number (double)` `boolean` `boolean` `boolean` `boolean` `number (double)` `Transformation`	Nothing	Scales the art item 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 (double)` `boolean` `boolean` `boolean` `boolean` `Transformation`	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the `angle` value is positive, clockwise if the value is negative.
`transform (transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])`	`Matrix` `boolean` `boolean` `boolean` `boolean` `number (double)` `Transformation`	Nothing	Transforms the art item by applying a transformation matrix.
`translate ([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])`	`number (double)` `number (double)` `boolean` `boolean` `boolean` `boolean`	Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.
`zOrder (zOrderCmd)`	`ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

Selecting paths in a document

```
// Selects all paths not part of a compound path

if ( app.documents.length > 0 ) {
    doc = app.activeDocument;
    count = 0;
    if ( doc.pathItems.length > 0 ) {
        thePaths = doc.pathItems;
        numPaths = thePaths.length;
        for ( i = 0; i < doc.pathItems.length; i++ ) {
            pathArt = doc.pathItems[i];
            if ( pathArt.parent.typename != "CompoundPathItem" ) {
                pathArt.selected = true;
                count++;
            }
        }
    }
}
```

Creating and modifying a compound path item

```
// Creates a new compound path item containing 3 path
// items, then sets the width and the color of the stroke
// to all items in the compound path

if ( app.documents.length > 0 ) {
    doc = app.activeDocument;
    newCompoundPath = doc.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 stroke and width properties of the compound path
    newPath.stroked = true;
    newPath.strokeWidth = 3.5;
    newPath.strokeColor = app.activeDocument.swatches[3].color;
}
```

CompoundPathItems

A collection of `CompoundPathItem` objects.

CompoundPathItem methods

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this collection (either a `Layer` or a `GroupItem`).
`typename`	`string`	Read-only. The class name of the referenced object.

CompoundPathItem methods

Method	Parameter type	Returns	What it does
`add ()`		`CompoundPathItem`	Creates a new `CompoundPathItem`.
`getByName (name)`	`string`	`CompoundPathItem`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`CompoundPathItem`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Counting compound paths

```
// Counts all compound path items in layer 1 of the current document

if ( app.documents.length > 0 ) {
    doc = app.activeDocument;
    numCompoundPaths = doc.layers[0].compoundPathItems.length;
}
```

Dataset

A set of data used for dynamic publishing. 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 item in order to create a dataset. See the class Variable.

Dataset properties

Property	Value type	What is it
`name`	`string`	Then name of the dataset.
`parent`	`Document`	Read-only. The name of the object that contains this dataset.
`typename`	`string`	Read-only. The class name of the referenced object.

Dataset methods

Method	Parameter type	Returns	What it does
`display ()`		Nothing	Displays the dataset.
`remove ()`		Nothing	Deletes this object.
`update ()`		Nothing	Updates the dataset.

Using variables and datasets

```
// Creates two variables, 1 visibility and 1 text,
// creates two datasets each with different values
// for the variables, then displays both datasets

var docRef = documents.add();

// Create visibility variable
var itemRef = docRef.pathItems.rectangle(600, 200, 150, 150);
var colorRef = new RGBColor;
colorRef.red = 255;
itemRef.fillColor = colorRef;
var visibilityVar = docRef.variables.add();
visibilityVar.kind = VariableKind.VISIBILITY;
itemRef.visibilityVariable = visibilityVar;

// Create text variable
var textRef = docRef.textFrames.add();
textRef.contents = "Text Variable, dataset 1";
textRef.top = 400;
textRef.left = 200;
var textVar = docRef.variables.add();
textVar.kind = VariableKind.TEXTUAL;
```

```
textRef.contentVariable = textVar;
redraw();

// Create dataset 1
var ds1 = docRef.dataSets.add();

// Change variable values and create dataset 2
itemRef.hidden = true;
textRef.contents = "Text Variable, dataset 2";
redraw();
var ds2 = docRef.dataSets.add();

// display each dataset
ds1.display();
redraw();
ds2.display();
redraw();
```

Datasets

A collection of `Dataset` objects.

Datasets properties

Property	Value type	What is it
`length`	`number`	Read-only. The number of datasets in the collection
`parent`	`Document`	Read-only. The name of the object that contains this dataset
`typename`	`string`	Read-only. Read-only. The class name of the referenced object.

Datasets methods

Method	Parameter type	Returns	What it does
`add ()`		`Dataset`	Creates a new dataset object.
`getByName (name)`	`string`	`Dataset`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`Dataset`	Gets an element from the collection.
`removeAll ()`		Nothing	Removes all elements in the collection.

Document

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

The 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. You cannot modify these properties in an existing document. See Application.open for more information on how document color spaces are handled.

Document properties

Property	Value type	What it is
`activeDataset`	`Dataset`	The currently opened dataset.
`activeLayer`	`Layer`	The active layer in the document.
`activeView`	`View`	Read-only. The document's current view.
`artboards`	`Artboards`	Read-only. All artboards in the document.
`brushes`	`Brushes`	Read-only. The brushes contained in the document.
`characterStyles`	`CharacterStyles`	Read-only. The list of character styles in this document.
`compoundPathItems`	`CompoundPathItems`	Read-only. The compound path items contained in the document.
`cropBox`	`array` of 4 numbers	The boundary of the document's cropping box for output, or `null` if no value has been set.
`cropStyle`	`CropOptions`	The style of the document's cropping box.
`dataSets`	`Datasets`	Read-only. The datasets contained in the document.
`defaultFillColor`	`Color`	The color to use to fill new paths if `defaultFilled` is `true`.
`defaultFilled`	`boolean`	If `true`, a new path should be filled.
`defaultFillOverprint`	`boolean`	If `true`, the art beneath a filled object should be overprinted by default.
`defaultStrokeCap`	`StrokeCap`	Default type of line capping for paths created.
`defaultStrokeColor`	`Color`	The stroke color for new paths if default stroked is `true`.
`defaultStroked`	`boolean`	If `true`, a new path should be stroked.

Property	Value type	What it is
`defaultStrokeDashes`	object	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 object, {}, for solid line.
`defaultStrokeDashOffset`	`number (double)`	The default distance into the dash pattern at which the pattern should be started for new paths.
`defaultStrokeJoin`	`StrokeJoin`	Default type of joints in new paths.
`defaultStrokeMiterLimit`	`number (double)`	When a default stroke join is set to `mitered`, this property specifies when the join will be converted to beveled (squared-off) by default. The default miter limit of 4 means that when the length of the point reaches four times the stroke weight, the join switches from a miter join to a bevel join. Range: 1 to 500; a value of 1 specifies a bevel join.
`defaultStrokeOverprint`	`boolean`	If `true`, the art beneath a stroked object should be overprinted by default.
`defaultStrokeWidth`	`number (double)`	Default width of stroke for new paths.
`documentColorSpace`	`DocumentColorSpace`	Read-only. The color specification system to use for this document's color space.
`fullName`	`File`	Read-only. The file associated with the document, which includes the complete path to the file.
`geometricBounds`	`array` of 4 `numbers`	Read-only. The bounds of the illustration excluding the stroke width of any objects in the document.
`gradients`	`Gradients`	Read-only. The gradients contained in the document.
`graphicStyles`	`GraphicStyles`	Read-only. The graphic styles defined in this document.
`graphItems`	`GraphItems`	Read-only. The graph art items in this document.
`groupItems`	`GroupItems`	Read-only. The group items contained in the document.
`height`	`number (double)`	Read-only. The height of the document.
`inkList`	object	Read-only. The list of inks in this document.
`kinsokuSet`	object	Read-only. The Kinsoku set of characters that cannot begin or end a line of Japanese text.

Property	Value type	What it is
layers	Layers	Read-only. The layers contained in the document.
legacyTextItems	LegacyTextItems	Read-only. The legacy text items in the document.
meshItems	MeshItems	Read-only. The mesh art items contained in the document.
mojikumiSet	object	Read-only. A list of names of predefined Mojikumi sets which specify the spacing for the layout and composition of Japanese text.
name	string	Read-only. The document's name (not the complete file path to the document).
nonNativeItems	NonNativeItems	Read-only. The non-native art items in this document.
outputResolution	number (double)	Read-only. The current output resolution for the document in dots per inch (dpi).
pageItems	PageItems	Read-only. The page items (all art item classes) contained in the document.
pageOrigin	array of 2 numbers	The zero-point of the page in the document without margins, relative to the overall height and width.
paragraphStyles	ParagraphStyles	Read-only. The list of paragraph styles in this document.
parent	Application	Read-only. The application that contains this document.
path	File	Read-only. The file associated with the document, which includes the complete path to the file.
pathItems	PathItems	Read-only. The path items contained in this document.
patterns	Patterns	Read-only. The patterns contained in this document.
placedItems	PlacedItems	Read-only. The placed items contained in this document.
pluginItems	PluginItems	Read-only. The plug-in items contained in this document.
printTiles	boolean	Read-only. If true, this document should be printed as tiled output.
rasterEffectSettings	RasterEffectOptions	The document's raster effect settings.

Property	Value type	What it is
rasterItems	RasterItems	Read-only. 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	RulerUnits	Read-only. The default measurement units for the rulers in the document.
saved	boolean	If true, the document has not been changed since last time it was saved.
selection	array of objects	References to the objects in this document's current selection, or null when nothing is selected. A reference to an insertion point is returned when there is an active insertion point in the contents of a selected 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.
showPlacedImages	boolean	Read-only. If true, placed images should be displayed in the document.
splitLongPaths	boolean	Read-only. If true, long paths should be split when printing.
spots	Spots	Read-only. The spot colors contained in this document.
stationery	boolean	Read-only. If true, the file is a stationery file.
stories	Stories	Read-only. The story items in this document.
swatches	Swatches	Read-only. The swatches in this document.
swatchGroups	SwatchGroups	Read-only. The swatch groups in this document.
symbolItems	SymbolItems	Read-only. The art items in the document linked to symbols.
symbols	Symbols	Read-only. The symbols in this document.
tags	Tags	Read-only. The tags in this document.
textFrames	TextFrameItems	Read-only. The text frames in this document.
tileFullPages	boolean	Read-only. If true, full pages should be tiled when printing this document.
typename	string	Read-only. Read-only. The class name of the referenced object.

Property	Value type	What it is
`useDefaultScreen`	`boolean`	Read-only. If `true`, the printer's default screen should be used when printing this document.
`variables`	`Variables`	Read-only. The variables defined in this document.
`variablesLocked`	`boolean`	If `true`, the variables are locked.
`views`	`Views`	Read-only. The views contained in this document.
`visibleBounds`	`array` of 4 numbers	Read-only. The visible bounds of the document, including stroke width of any objects in the illustration.
`width`	`number (double)`	Read-only. The width of this document.
`XMPString`	`string`	The XMP metadata packet associated with this document.

Document methods

Method	Parameter type	Returns	What it does
`activate ()`		Nothing	Brings the first window associated with the document to the front.
`arrange ([layoutStyle])`	`DocumentLayoutStyle`	Boolean	Arranges multiple documents in the given layout style.
`close ([saveOptions])`	`SaveOptions`	Nothing	Closes a document using specified save options. When you close a document, you should set your document reference to `null` to prevent your script from accidentally trying to access closed documents.

`convertCoordinate (coordinate, source, destination)`	`Point`
	`CoordinateSystem`
	`CoordinateSystem`
		`Point`	Converts the given point between artboard and document coordinate systems. Returns the converted point coordinates.

Method	Parameter type	Returns	What it does

`exportFile (exportFile, exportFormat [,options])`	`File`
	`ExportType`
	`variant`
		Nothing	Exports the document to the specified file using one of the predefined export file formats. The appropriate file extension is automatically appended to the file name, except for Photoshop® documents. For these, you must include the file extension (PSD) in the file specification.

`exportPDFPreset (file)`	`File`	Nothing	Exports the current PDF preset values to the file.
`exportPerspectiveGridPreset (file)`	`File`	Nothing	Exports the current perspective grid preset values to the file.
`exportPrintPreset (file)`	`File`	Nothing	Exports the current print preset values to the file.
`exportVariables (fileSpec)`	`File`	Nothing	Saves datasets into an XML library. The datasets contain variables and their associated dynamic data.
`fitArtboardToSelectedArt ([index])`	`number (long)`	boolean	Resizes the artboard at the given index to fit currently selected art. Index default is 0. Returns `true` on success.
`getPerspectiveActivePlane ()`		`PerspectiveGridPlaneType`	Retrieves the active plane of the active perspective grid of the document.
`hidePerspectiveGrid ()`		boolean	Hides the current active grid for the document. If no grid is visible, does nothing. Returns `true` if a grid is hidden.

`imageCapture (imageFile, [clipBounds], [options])`	`File`
	`Rect`
	`ImageCaptureOptions`
		Nothing	Captures the artwork content within the clipping boundaries in this document as a raster image, and writes the image data to a specified file.
			
			If the bounds parameter is omitted, captures the entire artwork.

`importCharacterStyles (fileSpec)`	`File`	Nothing	Loads the character styles from the Illustrator file.

Method	Parameter type	Returns	What it does
`importParagraphStyles (fileSpec)`	`File`	Nothing	Loads the paragraph styles from the Illustrator file.
`importPDFPreset (fileSpec [, replacingPreset])`	`File` `boolean`	Nothing	Loads all PDF presets from a file.
`importPerspectiveGridPreset (fileSpec [, perspectivePreset])`	`File` `String`	Nothing	Loads a specified perspective grid preset, or, if preset not specified, all presets from a file.
`importPrintPreset (printPreset, fileSpec)`	`string` `File`	Nothing	Loads the named print preset from the file.
`importVariables (fileSpec)`	`File`	Nothing	Imports a library containing datasets, variables, and their associated dynamic data. Importing variables overwrites existing variables and datasets.
`print ([options])`	`PrintOptions`	Nothing	Prints the document.
`rasterize (sourceArt [, clipBounds] [, options])`	`variant` `Rect` `RasterizeOptions`	`RasterItem`	Rasterizes the source art(s) within the specified clip bounds. The source art(s) is disposed of as a result of the rasterization.
`rearrangeArboards ([artboardLayout, artboardRowsOrCols, artboardSpacing, artboardMoveArtwork])`	`DocumentArtboardLayout` `integer` `Number` `boolean`	`boolean`	Rearranges artboards in the document. All arguments are optional. Default layout style is `DocumentArtboardLayout.GridByRow`. The second argument specifies the number of rows or columns, as appropriate for the chosen layout style, in the range `[1..docNumArtboards-1]`, or 1 (the default) for single row/column layouts. Spacing is a number of pixels, default 20. When last argument is true (the default), artwork is moved with the artboards.

Method	Parameter type	Returns	What it does
`save ()`		Nothing	Saves the document in it current location.
`saveAs (saveIn [, options])`	`File` `SaveOptions`	Nothing	Saves the document in the specified file as an Illustrator, EPS, or PDF file.
`selectObjectsOnActiveArtboard ()`		`boolean`	Selects the objects on the currently active artboard. Returns `true` on success.
`setActivePlane (gridPlane)`	`PerspectiveGridPlaneType`	`boolean`	Sets the active plane of the active perspective grid of the document. Returns `true` on success.
`selectPerspectivePreset (gridType, presetName)`	`PerspectiveGridType` `string`	boolean	Selects a predefined preset to define grid for the current document. Returns `true` on success.
`showPerspectiveGrid ()`		boolean	Shows the current active grid for the document, or if no grid is active, shows the default grid. Returns `true` on success.
`windowCapture (imageFile, windowSize)`	`File` `array` of 2 numbers	Nothing	Captures the current document window to the target TIFF image file.

Deselecting all objects in the current document

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

```
var docRef = activeDocument;

docRef.selection = null;
```

Closing a document

```
// Closes the active document without saving changes

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

Creating a document with defaults

```
// Creates a new document if none exists
```

```
// then sets fill and stroke defaults to true

if ( app.documents.length == 0 ) {
    doc = app.documents.add();
}
else {
    doc = app.activeDocument;
}
doc.defaultFilled = true;
doc.defaultStroked = true;
```

DocumentPreset

A preset document template to use when creating a new document. See `Documents.addDocument()`.

DocumentPreset properties

Property	Value type	What it is
`artboardLayout`	`DocumentArtboardLayout`	The layout of artboards in the new document. Default: `GridByRow`
`artboardRowsOrCols`	`number (long)`	The number of rows (for rows layout) or columns (for column layout) of artboards. Range: 1 to (`numArtboards` - 1) or 1 for single row or column layouts. Default: 1
`artboardSpacing`	`number (double)`	The spacing between artboards in the new document. Default: 20.0
`colorMode`	`DocumentColorSpace`	The color space for the new document.
`height`	`number (double)`	The height in document points. Default: 792.0
`numArtboards`	`number (long)`	The number of artboards for the new document. Range: 1 to 100. Default: 1
`previewMode`	`DocumentPreviewMode`	The preview mode for the new document.
`rasterResolution`	`DocumentRasterResolution`	The raster resolution for the new document.
`title`	`string`	The document title.
`transparencyGrid`	`DocumentTransparencyGrid`	The transparency grid color for the new document.
`typename`	`string`	Read-only. The class name of the referenced object.
`units`	`RulerUnits`	The ruler units for the new document.
`width`	`number (double)`	The width in document points. Default: 612.0

Documents

A collection of `Document` objects.

Documents properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

Documents methods

Method	Parameter type	Returns	What it does

`add`
`([documentColorSpace]`
`, width]`
`, height]`
`, numArtBoards]`
`, artboardLayout]`
`, artboardSpacing]`
`, artboardRowsOrCols])`
	`DocumentColorSpace`
	`number (double)`
	`number (double)`
	`number (long)`
	`DocumentArtboardLayout`
	`number (double)`
	`number (long)`
		`Document`	Creates a new document using optional parameters and returns a reference to the new document.

`addDocument`
`(startupPreset`
`, presetSettings]`
`, showOptionsDialog])`
	`string`
	`DocumentPreset`
	`boolean`
		`Document`	Creates a document from the preset, replacing any provided setting values, and returns a reference to the new document.

`getByName` `(name)`	`string`	`Document`	Gets the first element in the collection with the specified name.
`index` `(itemKey)`	`string, number`	`Document`	Gets an element from the collection.

Creating a new document

```
// Creates a new document with an RGB color space

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

EPSSaveOptions

Options for saving a document as an Illustrator EPS file, used with the saveAs method. All properties are optional.

EPSSaveOptions properties

Property	Value type	What it is
`artboardRange`	`string`	If `saveMultipleArtboards` is `true`, this is considered for multi-asset extraction, which specifies the artboard range. An empty string extracts all the artboards. Default: empty string
`cmykPostScript`	`boolean`	If `true`, use CMYK PostScript.
`compatibility`	`Compatibility`	Specifies the version of the EPS file format to save. Default: `Compatibility.ILLUSTRATOR1719`
`compatibleGradientPrinting`	`boolean`	If `true`, create a raster item of the gradient or gradient mesh so that PostScript Level 2 printers can print the object. Default: `false`
`embedAllFonts`	`boolean`	If `true`, all fonts used by the document should be embedded in the saved file (version 7 or later). Default: `false`
`embedLinkedFiles`	`boolean`	If `true`, linked image files are to be included in the saved document.
`flattenOuput`	`OutputFlattening`	How should transparency be flattened for file formats older than Illustrator 9.
`includeDocumentThumbnails`	`boolean`	If `true`, thumbnail image of the EPS artwork should be included.
`overprint`	`PDFOverprint`	Whether to preserve, discard, or simulate the overprint. Default: `PDFOverprint.PRESERVEPDFOVERPRINT`
`postScript`	`EPSPostScriptLevelEnum`	PostScript Language Level to use (Level 1 valid for file format version 8 or older). Default: `EPSPostScriptLevelEnum.LEVEL2`
`preview`	`EPSPreview`	The format for the EPS preview image.
`saveMultipleArtboards`	`boolean`	If `true`, all artboards or range of artboards are saved. Default: `false`
`typename`	`string`	Read-only. The class name of the referenced object.

Exporting to EPS format

```
// Exports current document to destFile as an EPS file with specified
// options, destFile contains the full path including the file name

function exportFileAsEPS (destFile) {
    var newFile = new File(destFile);
    var saveDoc;
    if ( app.documents.length == 0 )
        saveDoc = app.documents.add();
    else
        saveDoc = app.activeDocument;
    var saveOpts = new EPSSaveOptions();
    saveOpts.cmykPostScript = true;
    saveOpts.embedAllFonts = true;
    saveDoc.saveAs( newFile, saveOpts );
}
```

ExportOptionsAutoCAD

Options for exporting a document as an AutoCAD file, used with the exportFile method. All properties are optional.

When you export a document, a file extension is appended automatically. You should not include any file extension in the file specification. To override the default AutoCAD export format (DWG), use the exportFileFormat property.

ExportOptionsAutoCAD properties

Property	Value type	What it is
`alterPathsForAppearance`	`boolean`	If `true`, paths are altered if needed to maintain appearance. Default: `false`
`colors`	`AutoCADColors`	The colors exported into the AutoCAD file.
`convertTextToOutlines`	`boolean`	If `true`, text is converted to vector paths; preserves the visual appearance of type. Default: `false`
`exportFileFormat`	`AutoCADExportFileFormat`	The format to which the file is exported. Default: `AutoCADExportFileFormat.DWG`
`exportOption`	`AutoCADExportOption`	Specifies whether to preserve appearance or editability during export. Default: `AutoCADExportOption.MaximizeEditability`
`exportSelectedArtOnly`	`boolean`	If `true`, only selected artwork is exported. Default: `false`
`rasterFormat`	`AutoCADRasterFormat`	The format in which raster art is exported.
`scaleLineweights`	`boolean`	If `true`, line weights are scaled by the same scaling factor as the rest of the drawing. Default: `false`
`typename`	`string`	Read-only. The class name of the referenced object.
`unit`	`AutoCADUnit`	The measurement units from which to map.

Property	Value type	What it is
`unitScaleRatio`	`number (double)`	The ratio (as a percentage) by which output is scaled. Range: 0 to 1000
`version`	`AutoCADCompatibility`	The release of AutoCAD to which the file is exported. Default: `AutoCADCompatibility.AutoCADRelease24`

ExportOptionsFlash

Options for exporting a document as a Macromedia® Flash® (SWF) file, used with the exportFile method. All properties are optional.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsFlash properties

Property	Value type	What it is
`artClipping`	`ArtClippingOption`	How the art should be clipped during output. Default: `ArtClippingOption.OUTPUTARTBOUNDS`
`artboardRange`	`string`	If `saveMultipleArtboards` is `true`, this is considered for multi-asset extraction, which specifies the artboard range. An empty string extracts all the artboards. Default: empty string
`backgroundColor`	`RGBColor`	The background color of the exported Flash frames.
`backgroundLayers`	`array` of `Layer`	A list of layers to be included as the static background of the exported Flash frames.
`blendAnimation`	`BlendAnimationType`	The animation type for blended objects. Default: `BlendAnimationType.NOBLENDANIMATION`
`compressed`	`boolean`	If `true`, the exported file should be exported compressed. Default: `false`
`convertTextToOutlines`	`boolean`	If `true`, all text is converted to vector paths; preserves the visual appearance of type in all Flash players. Default: `false`
`curveQuality`	`number (long)`	The amount of curve information that should be presented. Default: 7
`exportAllSymbols`	`boolean`	If `true`, export all symbols defined in the palette. Default: `false`
`exportStyle`	`FlashExportStyle`	The style in which the exported data should be created in Flash. Default: `FlashExportStyle.ASFLASHFILE`
`exportVersion`	`FlashExportVersion`	The version of the exported SWF file. Default: `FlashExportVersion.FlashVersion9`.
`frameRate`	`number (double)`	The display rate in frames per second. Range: 0.01–120.0. Default: 12.0
`ignoreTextKerning`	`boolean`	If `true`, ignore kerning information in text objects. Default: `false`

Property	Value type	What it is
imageFormat	FlashImageFormat	How should the image in the exported Flash file be compressed. Default: FlashImageFormat.LOSSLESS
includeMetadata	boolean	If true, include minimal XMP metadata in the SWF file. Default: false
jpegMethod	FlashJPEGMethod	Specifies the JPEG method to use. Default: FlashJPEGMethod.Standard
jpegQuality	number (long)	Level of compression to use. Range 1 to 10. Default: 3
layerOrder	LayerOrderType	The order in which layers are exported to Flash frames. Default: LayerOrderType.BOTTOMUP
looping	boolean	If true, the Flash file is set to loop when run. Default: false
playbackAccess	FlashPlaybackSecurity	The access level for the exported SWF file. Default: FlashPlaybackSecurity.PlaybackLocal
preserveAppearance	boolean	If true, preserve appearance. If false, preserve editability. Default: false
readOnly	boolean	If true, export as read-only file. Default: false
replacing	SaveOptions	If a file with the same name already exists, should it be replaced. Default: SaveOptions.PROMPTTOSAVECHANGES
resolution	number (double)	The resolution in pixels per inch. Range: 72–2400. Default: 72
saveMultipleArtboards	boolean	If true, all artboards or range of artboards are saved. Default: false
typename	string	Read-only. The class name of the referenced object.

Exporting to Flash format

```
// Exports current document to destFile as a flash file with specified
// options, destFile contains the full path including the file name

function exportToFlashFile(destFile) {
    if ( app.documents.length > 0 ) {
        var exportOptions = new ExportOptionsFlash();
        var type = ExportType.FLASH;
        var fileSpec = new File(destFile);
        exportOptions.resolution = 150;
        app.activeDocument.exportFile( fileSpec, type, exportOptions );
    }
}
```

ExportOptionsGIF

Options for exporting a document as a GIF file, used with the exportFile method. All properties are optional.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsGIF properties

Property	Value type	What it is
`antiAliasing`	`boolean`	If `true`, the exported image should be anti-aliased. Default: `true`
`artBoardClipping`	`boolean`	If `true`, the exported image should be clipped to the art board. Default: `false`
`colorCount`	`number (long)`	The number of colors in the exported image's color table. Range: 2 to 256. Default: 128
`colorDither`	`ColorDitherMethod`	The method used to dither colors in the exported image. Default: `ColorDitherMethod.DIFFUSION`
`colorReduction`	`ColorReductionMethod`	The method used to reduce the number of colors in the exported image. Default: `ColorReductionMethod.SELECTIVE`
`ditherPercent`	`number (long)`	How much should the colors of the exported image be dithered, where 100.0 is 100%.
`horizontalScale`	`number (double)`	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0
`infoLossPercent`	`number (long)`	The level of information loss allowed during compression, where 100.0 is 100%.
`interlaced`	`boolean`	If `true`, the exported image should be interlaced. Default: `false`
`matte`	`boolean`	If `true`, the art board should be matted with a color. Default: `true`
`matteColor`	`RGBColor`	The color to use when matting the art board. Default: `WHITE`
`saveAsHTML`	`boolean`	If `true`, the exported image should be saved with an accompanying HTML file. Default: `false`
`transparency`	`boolean`	If `true`, the exported image should use transparency. Default: `true`
`typename`	`string`	Read-only. The class name of the referenced object.

Property	Value type	What it is
verticalScale	number (double)	The vertical scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0
webSnap	number (long)	How much should the color table be changed to match the web palette, where 100 is maximum. Default: 0

Exporting to GIF format

```
// Exports current document to dest as a GIF file with specified
// options, dest contains the full path including the file name

function exportToGIFFile(dest) {
    if ( app.documents.length > 0 ) {
        var exportOptions = new ExportOptionsGIF();
        var type = ExportType.GIF;
        var fileSpec = new File(dest);

        exportOptions.antiAliasing = false;
        exportOptions.colorCount = 64;
        exportOptions.colorDither = ColorDitherMethod.DIFFUSION;

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

ExportOptionsJPEG

Options for exporting a document as a JPEG file, used with the exportFile method. All properties are optional.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsJPEG properties

Property	Value type	What it is
`antiAliasing`	`boolean`	If `true`, the exported image should be anti-aliased. Default: `true`
`artBoardClipping`	`boolean`	If `true`, the exported image should be clipped to the art board.
`blurAmount`	`number (double)`	The amount of blur to apply to the exported image. Range: 0.0 to 2.0. Default: 0.0
`horizontalScale`	`number (double)`	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0
`matte`	`boolean`	If `true`, the art board should be matted with a color. Default: `true`
`matteColor`	`RGBColor`	The color to use when matting the art board. Default: `white`
`optimization`	`boolean`	If `true`, the exported image should be optimized for web viewing. Default: `true`
`qualitySetting`	`number (long)`	The quality of the exported image. Range: 0 to 100. Default: 30
`saveAsHTML`	`boolean`	If `true`, the exported image should be saved with an accompanying HTML file. Default: `false`
`typename`	`string`	Read-only. The class name of the referenced object.
`verticalScale`	`number (double)`	The vertical scaling factor to apply to the exported image. Range: 0.0 to 776.19. Default: 100.0

Exporting to JPEG format

```
// Exports current document to dest as a JPEG file with specified
// options, dest contains the full path including the file name

function exportFileToJPEG (dest) {
    if ( app.documents.length > 0 ) {
        var exportOptions = new ExportOptionsJPEG();
        var type = ExportType.JPEG;
        var fileSpec = new File(dest);
        exportOptions.antiAliasing = false;
        exportOptions.qualitySetting = 70;
        app.activeDocument.exportFile( fileSpec, type, exportOptions );
    }
}
```

ExportOptionsPhotoshop

Options for exporting a document as a Photoshop file, used with the exportFile method. All properties are optional.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsPhotoshop properties

Property	Value type	What it is
`antiAliasing`	`boolean`	If `true`, the exported image should be anti-aliased. Default: `true`
`artboardRange`	`string`	If `saveMultipleArtboards` is `true`, this is considered for multi-asset extraction, which specifies the artboard range. An empty string extracts all the artboards. Default: empty string
`editableText`	`boolean`	If `true`, text objects should be exported as editable text layers. Default: `true`
`embedICCProfile`	`boolean`	If `true`, an ICC profile should be embedded in the exported file. Default: `false`
`imageColorSpace`	`ImageColorSpace`	The color space of the exported file. Default: `ImageColorSpace.RGB`
`maximumEditability`	`boolean`	Preserve as much of the original document's structure as possible when exporting. Default: `true`
`resolution`	`number (double)`	Resolution of the exported file in dots per inch (dpi). Range: 72.0 to 2400.0. Default: 150.0
`saveMultipleArtboards`	`boolean`	If `true`, all artboards or range of artboards are saved. Default: `false`
`typename`	`string`	Read-only. The class name of the referenced object.
`warnings`	`boolean`	If `true`, a warning dialog should be displayed in case of conflicts in the export settings. Default: `true`
`writeLayers`	`boolean`	If `true`, the document layers should be presented in the exported document. Default: `true`

Exporting to Photoshop format

```
// Exports current document to dest as a PSD file with specified
// options, dest contains the full path including the file name

function exportFileToPSD (dest) {
    if ( app.documents.length > 0 ) {
        var exportOptions = new ExportOptionsPhotoshop();
        var type = ExportType.PHOTOSHOP;
        var fileSpec = new File(dest);
        exportOptions.resolution = 150;
        app.activeDocument.exportFile( fileSpec, type, exportOptions );
    }
}
```

ExportOptionsPNG8

Options for exporting a document as an 8-bit PNG file, used with the exportFile method. All properties are optional.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsPNG8 properties

Property	Value type	What it is
`antiAliasing`	`boolean`	If `true`, the exported image should be anti-aliased. Default: `true`
`artBoardClipping`	`boolean`	If `true`, the exported image should be clipped to the art board. Default: `false`
`colorCount`	`number (long)`	The number of colors in the exported image's color table. Range: 2 to 256. Default: 128
`colorDither`	`ColorDitherMethod`	The method used to dither colors in the exported image. Default: `ColorDitherMethod.Diffusion`
`colorReduction`	`ColorReductionMethod`	The method used to reduce the number of colors in the exported image. Default: `ColorReductionMethod.SELECTIVE`
`ditherPercent`	`number (long)`	The amount (as a percentage) that the colors of the exported image are dithered, where 100.0 is 100%. Range: 0 to 100. Default: 88
`horizontalScale`	`number (double)`	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0
`interlaced`	`boolean`	If `true`, the exported image should be interlaced. Default: `false`
`matte`	`boolean`	If `true`, the art board should be matted with a color. Default: `true`
`matteColor`	`RGBColor`	The color to use when matting the art board. Default: `white`
`saveAsHTML`	`boolean`	If `true`, the exported image be saved with an accompanying HTML file. Default: `false`
`transparency`	`boolean`	If `true`, the exported image use transparency. Default: `true`
`typename`	`string`	Read-only. The class name of the referenced object.

Property	Value type	What it is
verticalScale	number (double)	The vertical scaling factor to apply to the exported image, where 100.0 is 100. Default: 100.0
webSnap	number (long)	Specifies how much the color table should be changed to match the web palette, where 100 is maximum. Default: 0

Exporting to PNG8 format

```
// Exports current document to dest as a PNG8 file with specified
// options, dest contains the full path including the file name

function exportFileToPNG8 (dest) {
    if ( app.documents.length > 0 ) {
        var exportOptions = new ExportOptionsPNG8();
        var type = ExportType.PNG8;
        var fileSpec = new File(dest);
        exportOptions.colorCount = 8;
        exportOptions.transparency = false;
        app.activeDocument.exportFile( fileSpec, type, exportOptions );
    }
}
```

ExportOptionsPNG24

Options for exporting a document as a 24-bit PNG file, used with the exportFile method. All properties are optional.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsPNG24 properties

Property	Value type	What it is
`antiAliasing`	`boolean`	If `true`, the exported image be anti-aliased. Default: `true`
`artBoardClipping`	`boolean`	If `true`, the exported image be clipped to the art board. Default: `false`
`horizontalScale`	`number (double)`	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0
`matte`	`boolean`	If `true`, the art board be matted with a color. Default: `true`
`matteColor`	`RGBColor`	The color to use when matting the art board. Default: `white`
`saveAsHTML`	`boolean`	If `true`, the exported image be saved with an accompanying HTML file. Default: `false`
`transparency`	`boolean`	If `true`, the exported image use transparency. Default: `true`
`typename`	`string`	Read-only. The class name of the referenced object.
`verticalScale`	`number (double)`	The vertical scaling factor to apply to the exported image, where 100.0 is 100. Default: 100.0

Exporting to PNG24 format

```
// Exports current document to dest as a PNG24 file with specified
// options, dest contains the full path including the file name, saveAsHTML
// option creates an HTML version with the PNG file in an images folder

function exportFileToPNG24 (dest) {
    if ( app.documents.length > 0 ) {
        var exportOptions = new ExportOptionsPNG24();
        var type = ExportType.PNG24;
        var fileSpec = new File(dest);
        exportOptions.antiAliasing = false;
        exportOptions.transparency = false;
        exportOptions.saveAsHTML = true;
        app.activeDocument.exportFile( fileSpec, type, exportOptions );
    }
}
```

ExportOptionsSVG

Options for exporting a document as a SVG file, used with the exportFile method. All properties are optional.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsSVG properties

Property	Value type	What it is
`artboardRange`	`string`	A range of artboards to save, if `saveMultipleArtboards` is `true`. A comma-delimited list of artboard names., or the empty string to save all artboards. Default: empty string
`compressed`	`boolean`	If `true`, the exported file should be compressed. Default: `false`
`coordinatePrecision`	`number (long)`	The decimal precision for element coordinate values. Range: 1 to 7. Default: 3
`cssProperties`	`SVGCSSPropertyLocation`	How the CSS properties of the document should be included in the exported file. Default: `SVGCSSPropertyLocation.STYLEATTRIBUTES`
`documentEncoding`	`SVGDocumentEncoding`	How the text in the document should be encoded. Default: `SVGDocumentEncoding.ASCII`
`DTD`	`SVGDTDVersion`	The SVG version to which the file should conform. Default: `SVGDTDVersion.SVG1_1`
`embedRasterImages`	`boolean`	If `true`, the raster images contained in the document should be embedded in the exported file. Default: `false`
`fontSubsetting`	`SVGFontSubsetting`	Which font glyphs should be included in the exported file. Default: `SVGFontSubsetting.ALLGLYPHS`
`fontType`	`SVGFontType`	The type of font to included in the exported file. Default: `SVGFontType.CEFFONT`
`includeFileInfo`	`boolean`	If `true`, file information should be saved in the exported file. Default: `false`

Property	Value type	What it is
`includeUnusedStyles`	`boolean`	If `true`, save unused styles in the exported file. Default: `false`
`includeVariablesAndDatasets`	`boolean`	If `true`, variables and datasets should be saved in the exported file. Default: `false`
`optimizeForSVGViewer`	`boolean`	If `true`, the exported file should be optimized for the SVG Viewer. Default: `false`
`preserveEditability`	`boolean`	If `true`, Illustrator editing capabilities should be preserved when exporting the document. Default: `false`
`saveMultipleArtboards`	`boolean`	If `true`, save the artboards specified by `artboardRange` in the exported file. Default: `false`
`slices`	`boolean`	If `true`, slice data should be exported with the file. Default: `false`
`sVGAutoKerning`	`boolean`	If `true`, SVG automatic kerning is allowed in the file. Default: `false`
`sVGTextOnPath`	`boolean`	If `true`, the SVG `text-on-path` construct is allowed in the file. Default: `false`
`typename`	`string`	Read-only. The class name of the referenced object.

Exporting to SVG format

```
// Exports current document to dest as an SVG file with specified
// options, dest contains the full path including the file name

function exportFileToSVG (dest) {
    if ( app.documents.length > 0 ) {
        var exportOptions = new ExportOptionsSVG();
        var type = ExportType.SVG;
        var fileSpec = new File(dest);
        exportOptions.embedRasterImages = true;
        exportOptions.embedAllFonts = false;
        exportOptions.fontSubsetting = SVGFontSubsetting.GLYPHSUSED;
        app.activeDocument.exportFile( fileSpec, type, exportOptions );
    }
}
```

ExportOptionsTIFF

Options for exporting a document as a TIFF file, used with the exportFile method. All properties are optional.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsTIFF properties

Property	Value type	What it is
`antiAliasing`	`boolean`	If `true`, the exported image should be anti-aliased. Default: `true`
`artboardRange`	`string`	If `saveMultipleArtboards` is `true`, this is considered for multi-asset extraction, which specifies the artboard range. An empty string extracts all the artboards. Default: empty string
`byteOrder`	`TIFFByteOrder`	The byte order to use in the new file.
`imageColorSpace`	`ImageColorSpace`	The color space of the exported file. Default: `ImageColorSpace.RGB`
`IZWCompression`	`boolean`	If `true`, use IZW compression in the new file.
`resolution`	`number (double)`	Resolution of the exported file in dots per inch (dpi). Range: 72.0 to 2400.0. Default: 150.0
`saveMultipleArtboards`	`boolean`	If `true`, all artboards or range of artboards are saved. Default: `false`

Exporting to TIFF format

```
// Exports current document to dest as a TIFF file with specified
// options, dest contains the full path including the file name

function exportFileToPSD (dest) {
    if ( app.documents.length > 0 ) {
        var exportOptions = new ExportOptionsTIFF();
        var type = ExportType.TIFF;
        var fileSpec = new File(dest);
        exportOptions.resolution = 150;
        exportOptions.byteOrder = TIFFByteOrder.IBMPC;
        exportOptions.IZWCompression = false;
        app.activeDocument.exportFile( fileSpec, type, exportOptions );
    }
}
```

FXGSaveOptions

Specifies options which may be supplied when saving a document as an FXG file. All properties are optional.

FXGSaveOptions properties

Property	Value type	What it is
`artboardRange`	`string`	If `saveMultipleArtboards` is `true`, this is considered for multi-asset extraction, which specifies the artboard range. An empty string extracts all the artboards. Default: empty string
`blendsPolicy`	`BlendsExpandPolicy`	The policy used by FXG to expand blends. Default: `BlendsExpandPolicy.AUTOMATICALLYCONVERTBLENDS`
`downsampleLinkedImages`	`boolean`	If `true`, linked images are downsampled (at 72 dpi). Default: `false`
`filtersPolicy`	`FiltersPreservePolicy`	The policy used by FXG to preserve filters. Default: `FiltersPreservePolicy.KEEPFILTERSEDITABLE`
`gradientsPolicy`	`GradientsPreservePolicy`	The policy used by FXG to preserve gradients. Default: `GradientsPreservePolicy.AUTOMATICALLYCONVERTGRADIENTS`
`includeUnusedSymbols`	`boolean`	If `true`, unused symbols are included. Default: `false`
`preserveEditingCapabilities`	`boolean`	If `true`, the editing capabilities of FXG are preserved. Default: `true`
`saveMultipleArtboards`	`boolean`	If `true`, all artboards or range of artboards are saved. Default: `false`
`textPolicy`	`TextPreservePolicy`	The policy used by FXG to preserve text. Default: `TextPreservePolicy.AUTOMATICALLYCONVERTTEXT`
`version`	`FXGVersion`	The version of the FXG file format to create. Default `FXGVersion.VERSION2PT0`

Gradient

A gradient definition contained in a document. Scripts can create new gradients.

Gradient properties

Property	Value type	What it is
gradientStops	GradientStops	Read-only. The gradient stops contained in this gradient.
name	string	The gradient's name.
parent	Document	Read-only. The document that contains this gradient.
type	GradientType	The kind of the gradient, either radial or linear.
typename	string	Read-only. The class name of the referenced object.

Gradient methods

Method	Parameter type	Returns	What it does
remove ()		Nothing	Removes the referenced object from the document.

Creating and applying a gradient

```
// Creates a new gradient in current document then
// applies the gradient to the frontmost path item

if ( app.documents.length > 0 ) {
    // Create a color for both ends of the gradient
    var startColor = new RGBColor();
    var endColor = new RGBColor();

    startColor.red = 0;
    startColor.green = 100;
    startColor.blue = 255;
    endColor.red = 220;
    endColor.green = 0;
    endColor.blue = 100;

    // Create a new gradient
    // A new gradient always has 2 stops
    var newGradient = app.activeDocument.gradients.add();
    newGradient.name = "NewGradient";
    newGradient.type = GradientType.LINEAR;

    // Modify the first gradient stop
    newGradient.gradientStops[0].rampPoint = 30;
    newGradient.gradientStops[0].midPoint = 60;
    newGradient.gradientStops[0].color = startColor;
```

```
    // Modify the last gradient stop
    newGradient.gradientStops[1].rampPoint = 80;
    newGradient.gradientStops[1].color = endColor;

    // construct an Illustrator.GradientColor object referring to the
    // newly created gradient
    var colorOfGradient = new GradientColor();
    colorOfGradient.gradient = newGradient;

    // get first path item, apply new gradient as its fill
    var topPath = app.activeDocument.pathItems[0];
    topPath.filled = true;
    topPath.fillColor = colorOfGradient;
}
```

GradientColor

A gradient color specification in a `Gradient` object. A script can create a new gradient color using a reference to an existing gradient in the document. If no existing gradient object is referenced, a default gradient is supplied.

GradientColor properties

Property	Value type	What it is
`angle`	`number (double)`	The gradient vector angle in degrees. Default: 0.0
`gradient`	`Gradient`	Reference to the object defining the gradient.
`hiliteAngle`	`number (double)`	The gradient highlight vector angle in degrees.
`hiliteLength`	`number (double)`	The gradient highlight vector length.
`length`	`number (double)`	The gradient vector length.
`matrix`	`Matrix`	An additional transformation matrix to manipulate the gradient path.
`origin`	`array` of 2 numbers	The gradient vector origin, the center point of the gradient in this color.
`typename`	`string`	Read-only. The class name of the referenced object.

Changing a gradient stop color

```
// Creates a new RGB document, then changes the color
// of the first gradient stop of an indexed gradient

app.documents.add(DocumentColorSpace.RGB);

// Get a reference to the gradient that you want to change
var gradientRef = app.activeDocument.gradients[1];
// Create the new color
var startColor = new RGBColor();
startColor.red = 255;
startColor.green = 238;
startColor.blue = 98;
// apply new color to the first gradient stop
gradientRef.gradientStops[0].color = startColor;
```

Gradients

A collection of `Gradient` objects in a document.

Gradients properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

Gradients methods

Method	Parameter type	Returns	What it does
`add ()`		`Gradient`	Creates a new `Gradient` object.
`getByName (name)`	`string`	`Gradient`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`Gradient`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Removing a gradient

```
// Deletes the first gradient from the current document

if ( app.documents.length > 0 ) {
    app.activeDocument.gradients[0].remove();
}
```

GradientStop

A gradient stop definition that represents a point on a specific gradient defined in the document. Each gradient stop specifies a color change in the containing gradient. See Changing a gradient stop color for an example.

GradientStop properties

Property	Value type	What it is
`color`	`Color`	The color linked to this gradient stop.
`midPoint`	`number (double)`	The midpoint key value, specified as a percentage from 13.0 to 87.0.
`opacity`	`number (double)`	The opacity value for the gradient stop. Range: 0.0 to100.0
`parent`	`Gradient`	Read-only. The gradient that contains this gradient stop.
`rampPoint`	`number (double)`	The location of the color in the blend in a range from 0.0 to 100.0, where 100.0 is 100%.
`typename`	`string`	Read-only. The class name of the referenced object.

GradientStop methods

Method	Parameter type	Returns	What it does
`remove ()`		Nothing	Deletes this object.

GradientStops

A collection of `GradientStop` objects in a specific gradient. The elements are not named; you must access them by index.

GradientStops properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

GradientStops methods

Method	Parameter type	Returns	What it does
`add ()`		`GradientStop`	Creates a new object.
`getByName (name)`	`string`	`GradientStop`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`number`	`GradientStop`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all objects in this collection.

Adding a new gradient stop

```
// Adds a new gradient stop to a gradient, color of new stop is 70% gray

if ( app.documents.length > 0 && app.activeDocument.gradients.length > 0 ) {
    // Get a reference to the gradient to change
    var changeGradient = app.activeDocument.gradients[0];
    // Get a reference to the last gradient stop
    var origCount = changeGradient.gradientStops.length;
    var lastStop = changeGradient.gradientStops[origCount-1];
    // add the new gradient stop
    var newStop = changeGradient.gradientStops.add();

    // Set the values of the new gradient stop.
    // Move the original last gradient stop a bit to the left and
    // insert the new gradient stop at the old position
    newStop.rampPoint = lastStop.rampPoint;
    lastStop.rampPoint = lastStop.rampPoint - 10;
    // Create a new color to apply to the newly created gradient stop
    // --a Gray tint value of 70%
    var newStopColor = new GrayColor();
    newStopColor.gray = 70.0;
    newStop.color = newStopColor;
```

```
}
```

GraphicStyle

A graphic style. Each graphic style defines a set of appearance attributes that you can apply non-destructively to page items. Graphic styles are contained in documents. Scripts cannot create new graphic styles.

GraphicStyle properties

Property	Value type	What it is
`name`	`string`	The graphic style name.
`parent`	`Document`	Read-only. The document that contains this graphic style.
`typename`	`string`	Read-only. The class name of the referenced object.

GraphicStyle methods

Method	Parameter type	Returns	What it does
`applyTo (artItem)`	PageItem	Nothing	Applies this art style to a specified art item.
`mergeTo (artItem)`	PageItem	Nothing	Merges this art style into the current styles of a specified art item.
`remove ()`		Nothing	Deletes this object.

Applying a graphic style

```
// Duplicates each path item in the selection, places the duplicate into a
// new group, then applies a graphic style to the new groups items

if ( app.documents.length > 0 ) {
    var doc = app.activeDocument;
    var selected = doc.selection;

    var newGroup = doc.groupItems.add();
    newGroup.name = "NewGroup";
    newGroup.move( doc, ElementPlacement.PLACEATEND );

    var endIndex = selected.length;
    for ( i = 0; i < endIndex; i++ ) {
        if ( selected[i].typename == "PathItem" )
            selected[i].duplicate( newGroup, ElementPlacement.PLACEATEND );
    }
    for ( i = 0; i < newGroup.pageItems.length; i++ ) {
        doc.graphicStyles[1].applyTo( newGroup.pageItems[i] );
    }
}
```

GraphicStyles

A collection of `GraphicStyle` objects in a document.

GraphicStyles properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of graphic styles in the document.
`parent`	object	Read-only. The document that contains this graphic styles collection.
`typename`	`string`	Read-only. The class name of the referenced object.

GraphicStyles methods

Method	Parameter type:	Returns	What it does
`getByName (name)`	`string`	`GroupItem`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`GroupItem`	Gets an element from the collection.
`removeAll ()`		Nothing	Removes all elements in the referenced collection.

Counting graphics styles

```
//Counts the number of graphic styles in the active document
// and stores result in numberOfStyles

if ( app.documents.length > 0 ) {
    var numberOfStyles = app.activeDocument.graphicStyles.length;
}
```

GraphItem

Any graph artwork object. See example Rotating graph items below.

GraphItem properties

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout. You cannot set this value to `KnockoutState.Unknown`
`blendingMode`	`BlendModes`	The mode used when compositing an object.
`contentVariable`	`Variable`	The content variable bound to the graph item. It is not necessary to set the type of the `contentVariable` before binding. Illustrator automatically set the type to `GRAPH`.
`controlBounds`	`array` of 4 numbers	Read-only. The bounds of the object including stroke width and controls.
`editable`	`boolean`	Read-only. If `true`, this graph item is editable.
`geometricBounds`	`array` of 4 numbers	Read-only. The bounds of the object excluding stroke width.
`height`	`number (double)`	The height of the graph item.
`hidden`	`boolean`	If `true`, this graph item is hidden.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layer`	`Layer`	Read-only. The layer to which this graph item belongs.
`left`	`number`	The offset (in points) of the left side of the graph item from the left side of the page.
`locked`	`boolean`	If `true`, this graph item is locked.
`name`	`string`	The name of this graph item.
`note`	`string`	The note assigned to this item.
`opacity`	`number (double)`	The opacity of the object; the value is between 0.0 and 100.0.
`parent`	`Layer` or `GroupItem`	Read-only. The parent of this object.
`position`	`array` of 2 numbers	The position (in points) of the top left corner of the `graphItem` object in the format [x, y]. Does not include stroke weight.
`selected`	`boolean`	If `true`, this object is selected.
`sliced`	`boolean`	If `true`, the graph item is sliced. Default: `false`

Property	Value type	What it is
`tags`	`Tags`	Read-only. The tags contained in this graph item.
`top`	`number (double)`	The offset (in points) of the top of the graph item from the bottom of the page.
`typename`	`string`	Read-only. The type of the graph item.
`uRL`	`string`	The value of the Adobe URL tag assigned to this graph item.
`visibilityVariable`	`Variable`	The visibility variable bound to the graph item. It is not necessary to set the type of the `visibilityVariable` before binding. Illustrator automatically set the type to `VISIBILITY`.
`visibleBounds`	`array` of 4 numbers	Read-only. The visible bounds of the graph item including stroke width.
`width`	`number (double)`	The width of the graph item. Range: 0.0 to 16348.0
`wrapInside`	`boolean`	If `true`, the text frame object should be wrapped inside this object.
`wrapOffset`	`number (double)`	The offset to use when wrapping text around this object.
`wrapped`	`boolean`	If `true`, wrap text frame objects around this object. (Text frame must be above the object.)
`zOrderPosition`	`number (long)`	Read-only. The position of this art item within the stacking order of the group or layer (parent) that contains the art item.

GraphItem methods

Method	Parameter type	Returns	What it does
`duplicate ([relativeObject] [,insertionLocation])`	object `ElementPlacement`	`GraphItem`	Creates a duplicate of the selected object.
`move (relativeObject, insertionLocation)`	object `ElementPlacement`	`GraphItem`	Moves the object.
`remove ()`		Nothing	Deletes this object.

Method	Parameter type	Returns	What it does

`resize`
`(scaleX,`
`scaleY`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,scaleAbout])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Scales the art item 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 (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`Transformation`
		Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the `angle` value is positive, clockwise if the value is negative.

`transform`
`(transformationMatrix`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,transformAbout])`
	
	`Matrix`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Transforms the art item by applying a transformation matrix.

`translate`
`([deltaX]`
`[,deltaY]`
`[,transformObjects]`
`[,transformFillPatterns]`
`[,transformFillGradients]`
`[,transformStrokePatterns])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
		Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.

`zOrder` `(zOrderCmd)`	 `ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

GraphItems

A collection `GraphItems` objects, which gives you access to all the graph art items in an Illustrator document.

GraphItems properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

GraphItems methods

Method	Parameter type	Returns	What it does
`getByName (name)`	`string`	`GraphItems`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`GraphItems`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in the collection.

Rotating graph items

```
// Rotates each graph item in the current document 90 degrees.

// Verify a document with a graph item is open
var ok = false;
if (documents.length > 0) {
  var docRef = activeDocument
  var iCount = docRef.graphItems.length
  if( iCount > 0)  {
    ok = true;
    for (var i=0; i<iCount; i++) {
      var graphRef = docRef.graphItems[i];
      graphRef.selected = true;
      graphRef.rotate(90); //rotate clockwise 90 degrees
    }
    redraw();
  }
}
```

GrayColor

A grayscale color specification, used where a `color` object is required.

GrayColor properties

Property	Value type	What it is
`gray`	`number (double)`	The tint of the gray. Range: 0.0 to 100.0, where 0.0 is black and 100.0 is white.
`typename`	`string`	Read-only. The class name of the referenced object.

Changing a color to gray

```
// Sets the color of the first word in the active document
// to a shade of gray

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

    // Create the new color
    var textColor = new GrayColor();
    textColor.gray = 45;
    firstWord.filled = true;
    firstWord.fillColor = textColor;
}
```

GroupItem

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

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

GroupItem properties

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout.
`blendingMode`	`BlendModes`	The blend mode used when compositing an object.
`clipped`	`boolean`	If `true`, the group is clipped to the clipping mask.
`compoundPathItems`	`CompoundPathItems`	Read-only. The compound path items contained in this group.
`controlBounds`	`array` of 4 numbers	Read-only. The bounds of the object including stroke width and controls.
`editable`	`boolean`	Read-only. If `true`, this item is editable.
`geometricBounds`	`array` of 4 numbers	Read-only. The bounds of the object excluding stroke width.
`graphItems`	`GraphItems`	Read-only. The graph items contained in this group.
`groupItems`	`GroupItems`	Read-only. The group items contained in this group.
`height`	`number (double)`	The height of the group item.
`hidden`	`boolean`	If `true`, this group item is hidden.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layer`	`Layer`	Read-only. The layer to which this group item belongs.
`left`	`number (double)`	The position of the left side of the item (in points, measured from the left side of the page).
`legacyTextItems`	`LegacyTextItems`	Read-only. The legacy text items in the group.
`locked`	`boolean`	If `true`, this group item is locked.
`meshItems`	`MeshItems`	Read-only. The mesh items contained in this group.
`name`	`string`	The name of this group item.
`nonNativeItems`	`NonNativeItems`	Read-only. The non-native art items in this group.

Property	Value type	What it is
note	string	The note assigned to this item.
opacity	number (double)	The opacity of the object. Range: 0.0 to 100.0
pageItems	PageItems	Read-only. The page items (all art item classes) contained in this group.
parent	Layer or GroupItem	Read-only. The parent of this object.
pathItems	PathItems	Read-only. The path items contained in this group.
placedItems	PlacedItems	Read-only. The placed items contained in this group.
pluginItems	PluginItems	Read-only. The plug-in items contained in this group.
position	array of 2 numbers	The position (in points) of the top left corner of the groupItem object in the format [x, y]. Does not include stroke weight.
rasterItems	RasterItems	Read-only. The raster items contained in this group.
selected	boolean	If true, this group item is selected.
sliced	boolean	If true, the item sliced. Default: false
symbolItems	SymbolItems	Read-only. The symbol item objects in this group.
tags	Tags	Read-only. The tags contained in this group.
textFrames	TextFrameItems	Read-only. The text art items contained in this group.
top	number (double)	The position of the top of the item (in points, measured from the bottom of the page).
typename	string	Read-only. The class name of the referenced object.
uRL	string	The value of the Adobe URL tag assigned to this group item.
visibilityVariable	Variable	The visibility variable bound to the item.
visibleBounds	array of 4 numbers	Read-only. The visible bounds of the group item including stroke width.
width	number (double)	The width of the group item.
wrapInside	boolean	If true, the text frame object should be wrapped inside this object.
wrapOffset	number (double)	The offset to use when wrapping text around this object.
wrapped	boolean	If true, wrap text frame objects around this object (text frame must be above the object).
zOrderPosition	number (long)	Read-only. The position of this group object within the stacking order of the group or layer (parent) that contains the group object.

GroupItem methods

Method	Parameter type	Returns	What it does
`duplicate ([relativeObject] [,insertionLocation])`	object `ElementPlacement`	`GroupItem`	Creates a duplicate of the selected object.
`move (relativeObject, insertionLocation)`	object `ElementPlacement`	`GroupItem`	Moves the object.
`remove ()`		Nothing	Deletes this object.
`resize (scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])`	`number (double)` `number (double)` `boolean` `boolean` `boolean` `boolean` `number (double)` `Transformation`	Nothing	Scales the art item 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 (double)` `boolean` `boolean` `boolean` `boolean` `Transformation`	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the `angle` value is positive, clockwise if the value is negative.
`transform (transformationMatrix [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,transformAbout])`	`Matrix` `boolean` `boolean` `boolean` `boolean` `number (double)` `Transformation`	Nothing	Transforms the art item by applying a transformation matrix.
`translate ([deltaX] [,deltaY] [,transformObjects] [,transformFillPatterns] [,transformFillGradients] [,transformStrokePatterns])`	`number (double)` `number (double)` `boolean` `boolean` `boolean` `boolean`	Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.
`zOrder (zOrderCmd)`	`ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

Modifying all objects in a group

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.

```
// Creates a new group item, adds a new path item, of triangle shape, to the group, then
// adds a new text item to the group and sets the fill color of the text to red

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

    // Create a triangle and add text, the new art is created inside the group
    var trianglePath = triangleGroup.pathItems.add();
    trianglePath.setEntirePath( Array( Array(100, 100), Array(300, 100),
        Array(200, Math.tan(1.0471975) * 100 + 100) ) );
    trianglePath.closed = true;
    trianglePath.stroked = true;
    trianglePath.filled = false;
    trianglePath.strokeWidth = 3;

    var captionText = triangleGroup.textFrames.add();
    captionText.position = Array(100, 150);
    captionText.textRange.size = 48;
    captionText.contents = "A triangle";

    var fillColor = new RGBColor;
    fillColor.red = 255;
    fillColor.green = 0;
    fillColor.blue = 0;
    captionText.characters.fillColor = fillColor;
}
```

GroupItems

The collection of grouped art items in a document.

GroupItems properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

GroupItems methods

Method	Parameter type	Returns	What it does
`add ()`		`GroupItem`	Creates a new object.
`createFromFile (imageFile)`	`File`	`GroupItem`	Places an external vector art file as a group item in the document.
`getByName (name)`	`string`	`GroupItem`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`GroupItem`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Importing a PDF as a group item

The following script shows how you can import a PDF document using the `createFromFile` function. Before running this script you must create a one page PDF file and put it in the location `/temp/testfile1.pdf`.

```
// Embeds a new group item in to the current
// document from a file specified by dest
// dest should contain the full path and file name

function embedPDF(dest) {
    var embedDoc = new File(dest);
    if ( app.documents.length > 0 && embedDoc.exists ) {
        var doc = app.activeDocument;
        var placed = doc.groupItems.createFromFile( embedDoc );
    }
}
```

IllustratorSaveOptions

Options for saving a document as an Illustrator file, used with the saveAs method. All properties are optional.

IllustratorSaveOptions properties

Property	Value type	What it is
`artboardRange`	`string`	If `saveMultipleArtboards` is `true` (which is valid only for Illustrator 13 or earlier), the document is considered for multi-asset extraction, which specifies an artboard range. An empty string extracts all artboards. Default: empty string
`compatibility`	`Compatibility`	Specifies the version of Illustrator file format to create. Default: `Compatibility.ILLUSTRATOR19`
`compressed`	`boolean`	(Illustrator version 10 or later.) If `true`, the saved file is compressed. Default: `true`
`embedICCProfile`	`boolean`	(Illustrator version 9 or later.) If `true`, the document's ICC profile is embedded in the saved file. Default: `false`
`embedLinkedFiles`	`boolean`	(Illustrator version 7 or later.) If `true`, the linked image files is embedded in the saved file. Default: `false`
`flattenOutput`	`OutputFlattening`	(Versions before Illustrator 9.) How transparency should be flattened for older file format versions. Default: `OutputFlattening.PRESERVEAPPEARANCE`
`fontSubsetThreshold`	`number (double)`	(Illustrator version 9 or later.) Include a subset of fonts when less than this percentage of characters is used in the document. Range: 0.0 to 100.0. Default: 100.0
`pdfCompatible`	`boolean`	(Illustrator version 10 or later.) If `true`, the file is saved as a PDF compatible file. Default: `true`
`saveMultipleArtboards`	`boolean`	If `true`, all artboards or range of the artboards are saved. Valid for Illustrator 13 or earlier.
`typename`	`string`	Read-only. The class name of the referenced object.

Saving with options

```
// Saves the current document to dest as an AI file with specified options,
// dest specifies the full path and file name of the new file

function exportFileToAI (dest) {
    if ( app.documents.length > 0 ) {
        var saveOptions = new IllustratorSaveOptions();
        var ai8Doc = new File(dest);
        saveOptions.compatibility = Compatibility.ILLUSTRATOR8;
        saveOptions.flattenOutput = OutputFlattening.PRESERVEAPPEARANCE;
        app.activeDocument.saveAs( ai8Doc, saveOptions );
    }
}
```

ImageCaptureOptions

Options for image capture, used with the imageCapture method. All properties are optional.

ImageCaptureOptions properties

Property	Value type	What it is
`antiAliasing`	`boolean`	If `true`, the image result is anti-aliased. Default: `false`
`matte`	`boolean`	If `true`, the artboard is matted with a color. Default: `false`
`matteColor`	`RGBColor`	The color to use for the artboard matte. Default: `white`
`resolution`	`number (double)`	The resolution of the captured image file in points-per-inch (PPI), in the range [72.0 ... 2400.0]. Default: 150
`transparency`	`boolean`	If `true`, the image result is transparent. Default: `false`
`typename`	`string`	Read-only. The class name of the referenced object.

Ink

Associates a document ink name with ink information.

Ink properties

Property	Value type	What it is
`inkInfo`	`InkInfo`	The ink information
`name`	`string`	The ink's name
`typename`	`string`	Read-only. The class name of the object

InkInfo

Ink information for printing a document.

InkInfo properties

Property	Value type	What it is
angle	number (double)	The ink's screen angle in degrees. Range: -360 to 360
customColor	Color	The color of the custom ink.
density	number (double)	The neutral density. Minimum: 0.0
dotShape	string	The dot shape name.
frequency	number (double)	The ink's frequency. Range: 0.0 to 1000.0
kind	InkType	The ink type.
printingStatus	InkPrintStatus	The ink printing status.
trapping	TrappingType	The trapping type.
trappingOrder	number (long)	The order of trapping for the ink. Range: 1 to 4 for CMYK
typename	string	Read-only. The class name of the object.

Getting ink information

```
// Displays the current documents inks in a text frame

var docRef = documents.add();
var textRef = docRef.textFrames.add();

// assemble a string of the inks in this document
var sInks = "";
var iLength = activeDocument.inkList.length;

for(var i=0; i<iLength; i++) {
    sInks += docRef.inkList[i].name;
    sInks += "\r\t";
    sInks += "Frequency = " + docRef.inkList[i].inkInfo.frequency;
    sInks += "\r\t";
    sInks += "Density = " + docRef.inkList[i].inkInfo.density;
    sInks += "\r";
}
textRef.contents = sInks;
textRef.top = 600;
textRef.left = 200;
redraw();
```

InsertionPoint

A location between characters that is used to insert new text objects. An insertion point is contained in an `InsertionPoints` collection.

InsertionPoint properties

Property	Value type	What it is
`characters`	`Characters`	Read-only. All the characters in this text range.
`lines`	`Lines`	Read-only. All the lines in this text range.
`paragraphs`	`Paragraphs`	Read-only. All the paragraphs in this text range.
`parent`	`TextRange`	Read-only. The object's container.
`story`	`Story`	Read-only. The story to which the text range belongs.
`textRanges`	`TextRanges`	Read-only. All of the text in this text range.
`typename`	`string`	Read-only. The class name o f the object.
`words`	`Words`	Read-only. All the words contained in this text range.

InsertionPoints

A collection of `InsertionPoint` objects.

InsertionPoints properties

Property	Value type	What it is
`length`	`number`	Read-only. Number of elements in the collection.
`parent`	object	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the object.

InsertionPoints methods

Method	Parameter type	Returns	What it does
`index (itemKey)`	`string, number`	`InsertionPoint`	Gets an element from the collection.

Using insertion points to add spaces

```
// Creates a new document, adds text then inserts a
// space between each character using insertion points

var docRef = documents.add();
var textRef = docRef.textFrames.add();
textRef.contents = "Wouldn't you rather be scripting?";
textRef.top = 400;
textRef.left = 100;
textRef.textRange.characterAttributes.size = 20;
redraw();

// Add a space between each character using insertion points.
var ip;
for(var i=0; i<(textRef.insertionPoints.length); i+=2) {
  ip = textRef.insertionPoints[i];
  ip.characters.add(" ");
}
```

LabColor

A color specification in the CIE Lab color space, used where a `color` object is required.

LabColor properties

Property	Value type	What it is
`a`	`number (double)`	The a (red-green) color value. Range -128.0–128.0. Default: 0.0
`b`	`number (double)`	The b (yellow-blue) color value. Range -128.0–128.0. Default: 0.0
`l`	`number (double)`	The l (lightness) color value. Range -128.0–128.0. Default: 0.0

Layer

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

The `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.

Layer properties

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout. You cannot set this value to `KnockoutState.Unknown`.
`blendingMode`	`BlendModes`	The mode used when compositing an object.
`color`	`RGBColor`	The layer's selection mark color.
`compoundPathItems`	`CompoundPathItems`	Read-only. The compound path items contained in this layer.
`dimPlacedImages`	`boolean`	If `true`, placed images should be rendered as dimmed in this layer.
`graphItems`	`GraphItems`	Read-only. The graph items contained in this layer.
`groupItems`	`GroupItems`	Read-only. The group items contained in this layer.
`hasSelectedArtwork`	`boolean`	If `true`, an object in this layer has been selected; set to `false` to deselect all objects in the layer.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layers`	`Layers`	Read-only. The layers contained in this layer.
`legacyTextItems`	`LegacyTextItems`	Read-only. The legacy text items in this layer.
`locked`	`boolean`	If `true`, this layer is editable; set to `false` to lock the layer.
`meshItems`	`MeshItems`	Read-only. The mesh items contained in this layer.
`name`	`string`	The name of this layer.
`nonNativeItems`	`NonNativeItems`	The non-native art items in this layer.
`opacity`	`number (double)`	The opacity of the layer. Range: 0.0 to 100.0
`pageItems`	`PageItems`	Read-only. The page items (all art item classes) contained in this layer.
`parent`	`Document` or `Layer`	Read-only. The document or layer that contains this layer.

Property	Value type	What it is
pathItems	PathItems	Read-only. The path items contained in this layer.
placedItems	PlacedItems	Read-only. The placed items contained in this layer.
pluginItems	PluginItems	Read-only. The plug-in items contained in this layer.
preview	boolean	If true, this layer should be displayed using preview mode.
printable	boolean	If true, this layer should be printed when printing the document.
rasterItems	RasterItems	Read-only. The raster items contained in this layer.
sliced	boolean	If true, the layer item is sliced. Default: false
symbolItems	SymbolItems	Read-only. The symbol items contained in the layer.
textFrames	TextFrameItems	Read-only. The text art items contained in this layer.
typename	string	Read-only. The class name of the referenced object.
visible	boolean	If true, this layer is visible.
zOrderPosition	number (long)	Read-only. The position of this layer within the stacking order of layers in the document.

Layer methods

Method	Parameter type	Returns	What does it do
move (relativeObject, insertionLocation)	object ElementPlacement	Layer	Moves the object.
remove ()		Nothing	Deletes this object.
zOrder (ZOrderCmd)	ZOrderMethod	Nothing	Arranges the layer's position in the stacking order of the containing layer or document (parent) of this object

Bringing a layer to the front

```
// Moves the bottom layer to become the topmost layer

if (documents.length > 0) {
    countOfLayers = activeDocument.layers.length;
    if (countOfLayers > 1) {
        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.

Layers properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

Layers methods

Method	Parameter type	Returns	What it does
`add ()`		`Layer`	Creates a new layer in the document.
`getByName (name)`	`string`	`Layer`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`Layer`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Finding and deleting layers

```
// Deletes all layers whose name begins with "Temp" in all open documents

// loop through all open documents
var layersDeleted = 0;
for ( i = 0; i < app.documents.length; i++ ) {
    var targetDocument = app.documents[i];
    var layerCount = targetDocument.layers.length;
    // Loop through layers from the back, to preserve index
    // of remaining layers when we remove one
    for (var ii = layerCount - 1; ii >= 0; ii-- ) {
        targetLayer = targetDocument.layers[ii];
        var layerName = new String( targetLayer.name );
        if ( layerName.indexOf("Temp") == 0 ) {
            targetDocument.layers[ii].remove();
            layersDeleted++;
        }
    }
}
```

LegacyTextItem

A text object created in Illustrator CS (version 10) or earlier, which is uneditable until converted. To convert legacy text, see convertToNative.

You can view, move, and print legacy text, but you cant edit it. Legacy text has an "x" through its bounding box when selected.

LegacyTextItem properties

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout.
`blendingMode`	`BlendModes`	The blend mode used when compositing an object.
`controlBounds`	`array` of 4 numbers	Read-only. The bounds of the object including stroke width and controls.
`converted`	`boolean`	Read-only. If `true`, the legacy text item has been updated to a native text frame item.
`editable`	`boolean`	Read-only. If `true`, this item is editable.
`geometricBounds`	`array` of 4 numbers	Read-only. The bounds of the object excluding stroke width.
`height`	`number (double)`	The height of the group item.
`hidden`	`boolean`	If `true`, this item is hidden.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layer`	`Layer`	Read-only. The layer to which this item belongs.
`left`	`number (double)`	The position of the left side of the item (in points, measured from the left side of the page).
`locked`	`boolean`	If `true`, this item is locked.
`name`	`string`	The name of this item.
`note`	`string`	The note assigned to this item.
`opacity`	`number (double)`	The opacity of the object. Range: 0.0 to 100.0
`parent`	`Layer` or `GroupItem`	Read-only. The parent of this object.
`position`	`array` of 2 numbers	The position (in points) of the top left corner of the `legacyTextItem` object in the format [x, y]. Does not include stroke weight.
`selected`	`boolean`	If `true`, this item is selected.
`sliced`	`boolean`	If `true`, the item sliced. Default: `false`
`tags`	`Tags`	Read-only. The tags contained in this item.

Property	Value type	What it is
top	number (double)	The position of the top of the item (in points, measured from the bottom of the page).
typename	string	Read-only. The class name of the referenced object.
uRL	string	The value of the Adobe URL tag assigned to this item.
visibilityVariable	Variable	The visibility variable bound to the item.
visibleBounds	array of 4 numbers	Read-only. The visible bounds of the item including stroke width.
width	number (double)	The width of the item.
wrapInside	boolean	If true, the text frame object should be wrapped inside this object.
wrapOffset	number (double)	The offset to use when wrapping text around this object.
wrapped	boolean	If true, wrap text frame objects around this object (text frame must be above the object).
zOrderPosition	number (long)	Read-only. The position of this item within the stacking order of the group or layer (parent) that contains the item.

LegacyTextItem methods

Method	Parameter type	Returns	What it does
convertToNative ()		GroupItem	Converts the legacy text item to a text frame and deletes the original legacy text.
duplicate ([relativeObject] [,insertionLocation])	object ElementPlacement	LegacyTextItem	Creates a duplicate of the selected object.
move (relativeObject, insertionLocation)	object ElementPlacement	LegacyTextItem	Moves the object.
remove ()		Nothing	Deletes this object.

Method	Parameter type	Returns	What it does

`resize`
`(scaleX,`
`scaleY`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,scaleAbout])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Scales the art item 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 (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`Transformation`
		Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the `angle` value is positive, clockwise if the value is negative.

`transform`
`(transformationMatrix`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,transformAbout])`
	
	`Matrix`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Transforms the art item by applying a transformation matrix.

`translate`
`([deltaX]`
`[,deltaY]`
`[,transformObjects]`
`[,transformFillPatterns]`
`[,transformFillGradients]`
`[,transformStrokePatterns])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
		Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.

`zOrder` `(zOrderCmd)`	 `ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

LegacyTextItems

A collection of `LegacyTextItem` objects.

LegacyTextItems properties

Property	Value type	What it is
`length`	`number`	Read-only. Number of elements in the collection.
`parent`	object	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the object.

LegacyTextItems methods

Method	Parameter type	Returns	What it does
`convertToNative ()`		`boolean`	Creates text frames from all legacy text items; the original legacy text items are deleted. Returns `true` on success.
`getByName (name)`	`string`	`LegacyTextItem`	Get the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`LegacyTextItem`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Lines

A collection of `TextRange` objects representing lines of text in a text frame. The elements are not named; you must access them by index.

Lines properties

Property	Value type	What it is
`length`	`number`	Read-only. Number of elements in the collection.
`parent`	object	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the object.

Lines methods

Method	Parameter type	Returns	What it does
`index (itemKey)`	`number`	`TextRange`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Matrix

A transformation matrix specification, used to transform the geometry of objects. Use it to specify and retrieve 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 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.

Matrix properties

Property	Value type	What it is
`mValueA`	`number (double)`	Matrix property `a`.
`mValueB`	`number (double)`	Matrix property `b`.
`mValueC`	`number (double)`	Matrix property `c`.
`mValueD`	`number (double)`	Matrix property `d`.
`mValueTX`	`number (double)`	Matrix property `tx`.
`mValueTY`	`number (double)`	Matrix property `ty`.
`typename`	`string`	Read-only. The class name of the referenced object.

Combining matrices to apply multiple transformations

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.

```
// Tranforms all art in a document using translation and rotation matrices,
// moves art half an inch to the right and 1.5 inches up on the page

if ( app.documents.length > 0 ) {
    var moveMatrix = app.getTranslationMatrix( 0.5, 1.5 );
    // Add a rotation to the translation, 10 degrees counter clockwise
    var totalMatrix = concatenateRotationMatrix( moveMatrix, 10 );
    // apply the transformation to all art in the document
    var doc = app.activeDocument;
    for ( i = 0; i < doc.pageItems.length; i++ ) {
        doc.pageItems[i].transform( totalMatrix );
    }
}
```

MeshItem

A gradient mesh art item. You cannot create mesh items from a script. However, you can copy an existing mesh item with the `duplicate` method, then use the one of the move methods to place the copy at the proper location.

MeshItem properties

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout.
`blendingMode`	`BlendModes`	The blend mode used when compositing an object.
`controlBounds`	`array` of 4 numbers	Read-only. The bounds of the object including stroke width and controls.
`editable`	`boolean`	Read-only. If `true`, this item is editable.
`geometricBounds`	`array` of 4 numbers	Read-only. The bounds of the object excluding stroke width.
`height`	`number (double)`	The height of the group item.
`hidden`	`boolean`	If `true`, this item is hidden.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layer`	`Layer`	Read-only. The layer to which this item belongs.
`left`	`number (double)`	The position of the left side of the item (in points, measured from the left side of the page).
`locked`	`boolean`	If `true`, this item is locked.
`name`	`string`	The name of this item.
`note`	`string`	The note assigned to this item.
`opacity`	`number (double)`	The opacity of the object. Range: 0.0 to 100.0
`parent`	`Layer` or `GroupItem`	Read-only. The parent of this object.
`position`	`array` of 2 numbers	The position (in points) of the top left corner of the `meshItem` object in the format [x, y]. Does not include stroke weight.
`selected`	`boolean`	If `true`, this item is selected.
`sliced`	`boolean`	If `true`, the item sliced. Default: `false`
`tags`	`Tags`	Read-only. The tags contained in this item.
`top`	`number (double)`	The position of the top of the item (in points, measured from the bottom of the page).

Property	Value type	What it is
typename	string	Read-only. The class name of the referenced object.
uRL	string	The value of the Adobe URL tag assigned to this item.
visibilityVariable	Variable	The visibility variable bound to the item.
visibleBounds	array of 4 numbers	Read-only. The visible bounds of the item including stroke width.
width	number (double)	The width of the item.
wrapInside	boolean	If true, the text frame object should be wrapped inside this object.
wrapOffset	number (double)	The offset to use when wrapping text around this object.
wrapped	boolean	If true, wrap text frame objects around this object (text frame must be above the object).
zOrderPosition	number (long)	Read-only. The position of this item within the stacking order of the group or layer (parent) that contains the item.

MeshItem methods

Method	Parameter type	Returns	What it does
duplicate ([relativeObject] [,insertionLocation])	 object ElementPlacement	MeshItem	Creates a duplicate of the selected object.
move (relativeObject, insertionLocation)	 object ElementPlacement	MeshItem	Moves the object.
remove ()		Nothing	Deletes this object.
resize (scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])	 number (double) number (double) boolean boolean boolean boolean number (double) Transformation	Nothing	Scales the art item 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 (double) boolean boolean boolean boolean Transformation	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the angle value is positive, clockwise if the value is negative.

Method	Parameter type	Returns	What it does

`transform`
`(transformationMatrix`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,transformAbout])`
	
	`Matrix`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Transforms the art item by applying a transformation matrix.

`translate`
`([deltaX]`
`[,deltaY]`
`[,transformObjects]`
`[,transformFillPatterns]`
`[,transformFillGradients]`
`[,transformStrokePatterns])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
		Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.

`zOrder` `(zOrderCmd)`	 `ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

Finding and locking mesh items

```
// Locks all mesh items in the current document

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

MeshItems

A collection of `MeshItem` objects.

MeshItems properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection
`parent`	object	Read-only. The parent of this object
`typename`	`string`	Read-only. The class name of the referenced object.

MeshItems methods

Method	Parameter type	Returns	What it does
`getByName (name)`	`string`	`MeshItem`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`MeshItem`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Copying mesh items to another document

To run this script, 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.

```
// Copies all mesh items from one document to a new document

if ( app.documents.length > 0 ) {
    var srcDoc = documents[0];
    var locationOffset = 0;
    var targetDoc = documents.add();

    for ( i = 0; i < srcDoc.meshItems.length; i++) {
        srcItem = srcDoc.meshItems[i];
        var dupItem = srcDoc.meshItems[i].duplicate( targetDoc,
            ElementPlacement.PLACEATEND );

        // offset the copied items' position on the y axis
        dupItem.position = Array( 100, 50 + locationOffset );
        locationOffset += 50;
    }
}
```

NoColor

Represents the "none" color. Assigning a `NoColor` object to the fill or stroke color of an art item is equivalent to setting the `filled` or `stroked` property to `false`.

NoColor properties

Property	Value type	What it is
`typename`	`string`	Read-only. The class name of the object

Using NoColor to remove a fill color

```
// Creates 2 overlapping objects with different fill colors.
// Assign the top object a fill color of "NoColor"
// allowing the bottom object to become visible.

// create 2 overlapping objects one blue, one red;
var docRef = documents.add();
var itemRef1 = docRef.pathItems.rectangle(500, 200, 200, 100);
var itemRef2 = docRef.pathItems.rectangle(550, 150, 200, 200);
var rgbColor = new RGBColor();
rgbColor.red = 255;
itemRef2.fillColor = rgbColor;
rgbColor.blue = 255;
rgbColor.red = 0;
itemRef1.fillColor = rgbColor;
redraw();

// create a nocolor and assign it to the top object
var noColor = new NoColor();
itemRef2.fillColor = noColor;
redraw();
```

NonNativeItem

A non-native artwork item.

NonNativeItem properties

These classes inherit all properties from the `page item` class.

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout.
`blendingMode`	`BlendModes`	The blend mode used when compositing an object.
`controlBounds`	`array` of 4 numbers	Read-only. The bounds of the object including stroke width and controls.
`editable`	`boolean`	Read-only. If `true`, this item is editable.
`geometricBounds`	`array` of 4 numbers	Read-only. The bounds of the object excluding stroke width.
`height`	`number (double)`	The height of the group item.
`hidden`	`boolean`	If `true`, this item is hidden.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layer`	`Layer`	Read-only. The layer to which this item belongs.
`left`	`number (double)`	The position of the left side of the item (in points, measured from the left side of the page).
`locked`	`boolean`	If `true`, this item is locked.
`name`	`string`	The name of this item.
`note`	`string`	The note assigned to this item.
`opacity`	`number (double)`	The opacity of the object. Range: 0.0 to 100.0
`parent`	`Document`, `Layer`, or `GroupItem`	Read-only. The parent of this object.
`position`	`array` of 2 numbers	The position (in points) of the top left corner of the `NonNativeItem` object in the format [x, y]. Does not include stroke weight.
`selected`	`boolean`	If `true`, this item is selected.
`sliced`	`boolean`	If `true`, the item sliced. Default: `false`
`tags`	`Tags`	Read-only. The tags contained in this item.
`top`	`number (double)`	The position of the top of the item (in points, measured from the bottom of the page).

Property	Value type	What it is
`typename`	`string`	Read-only. The class name of the referenced object.
`uRL`	`string`	The value of the Adobe URL tag assigned to this item.
`visibilityVariable`	`Variable`	The visibility variable bound to the item.
`visibleBounds`	`array` of 4 numbers	Read-only. The visible bounds of the item including stroke width.
`width`	`number (double)`	The width of the item.
`wrapInside`	`boolean`	If `true`, the non-native-item object should be wrapped inside this object.
`wrapOffset`	`number (double)`	The offset to use when wrapping text around this object.
`wrapped`	`boolean`	If `true`, wrap non-native-item objects around this object (non-native-item object must be above the object).
`zOrderPosition`	`number`	Read-only. The position of this item within the stacking order of the group or layer (`parent`) that contains the item.

NonNativeItem methods

Method	Parameter type	Returns	What it does
`duplicate ([relativeObject] [,insertionLocation])`	object `ElementPlacement`	`NonNativeItem`	Creates a duplicate of the selected object.
`move (relativeObject, insertionLocation)`	object `ElementPlacement`	`NonNativeItem`	Moves the object.
`remove ()`		Nothing	Deletes this object.
`removeAll ()`		Nothing	Deletes all elements in this collection.
`resize (scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])`	`number (double)` `number (double)` `boolean` `boolean` `boolean` `boolean` `number (double)` `Transformation`	Nothing	Scales the art item 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 (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`Transformation`
		Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the `angle` value is positive, clockwise if the value is negative.

`transform`
`(transformationMatrix`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,transformAbout])`
	
	`Matrix`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Transforms the art item by applying a transformation matrix.

`translate`
`([deltaX]`
`[,deltaY]`
`[,transformObjects]`
`[,transformFillPatterns]`
`[,transformFillGradients]`
`[,transformStrokePatterns])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
		Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.

`zOrder` `(zOrderCmd)`	 `ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

NonNativeItems

A collection of `NonNativeItem` objects.

NonNativeItems properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

NonNativeItems methods

Method	Parameter type	Returns	What it does
`getByName (name)`	`string`	`NonNativeItem`, `SymbolItem`	Gets the first element in the collection with the specified name.

OpenOptions

Options for opening a document, used with the open method.

OpenOptions properties

Property	Value type	What it is
`convertCropAreaToArboard`	`boolean`	Optional. Convert crop areas to artboards when opening a legacy document in Illustrator CS4 or later. When false, crop areas are discarded. Default: true.
`convertTilesToArboard`	`boolean`	Optional. Convert print tiles to artboards when opening a legacy document in Illustrator CS4 or later. Default: false.
`createArtboardWithArtworkBoundingBox`	`boolean`	Optional. Create an artboard with the dimensions of the bounding box of the artwork when opening a legacy document in Illustrator CS4 or later. Default: false.
`openAs`	`LibraryType`	Optional. Open the file as an Illustrator library of this type. Default: `LibraryType.IllustratorArtwork`.
`preserveLegacyArtboard`	`boolean`	Optional. Preserve legacy artboards when opening a legacy document in Illustrator CS4 or later. Default: true.
`updateLegacyGradientMesh`	`boolean`	If `true`, preserves the spot colors in the gradient mesh objects for legacy documents (pre-Illustrator CS4). Default: `true`
`updateLegacyText`	`boolean`	Optional. If `true`, update all legacy text items (from previous versions of Illustrator). Default: `false`

Automatically updating legacy text on open

```
// Opens a file with legacy text (AI 10 or older), using
// OpenOptions to automatically update the legacy text.

var fileRef = filePath;
if (fileRef != null) {
    var optRef = new OpenOptions();
    optRef.updateLegacyText = true;
    var docRef = open(fileRef, DocumentColorSpace.RGB, optRef);
}
```

OpenOptionsAutoCAD

Options for opening an AutoCAD drawing, used with the open method.

OpenOptionsAutoCAD properties

Property	Value type	What it is
`centerArtwork`	`boolean`	If `true`, the artwork is centered on the artboard. Default: `true`
`globalScaleOption`	`AutoCADGlobalScaleOption`	How to scale the drawing on import. Default: `AutoCADGlobalScaleOption.FitArtboard`
`globalScalePercent`	`double`	The value when `globalScaleOption` is `AutoCADGlobalScaleOption.ScaleByValue`, expressed as a percentage. Range: 0.0 to 100.0. Default is 100.0
`mergeLayers`	`boolean`	If `true`, the layers of the artwork are merged. Default: `false`
`parent`	`object`	Read-only. The object's container.
`scaleLineweights`	`boolean`	If `true`, line weights are scaled by the same factor as the rest of the drawing. Default: `false`
`selectedLayoutName`	`string`	The name of the layout in the drawing to import.
`typename`	`string`	Read-only. The class name of the object.
`unit`	`AutoCADUnit`	The unit to map to. Default: `AutoCADUnit.Millimeters`
`unitScaleRatio`	`double`	The ratio by which to scale while mapping units. Default: 1.0

OpenOptionsFreeHand

Options for opening a FreeHand file.

OpenOptionsFreeHand properties

Property	Value type	What it is
`convertTextToOutlines`	`boolean`	If `true`, all text is converted to vector paths; preserves the visual appearance of type. Default: `false`
`importSinglePage`	`boolean`	If `true`, imports only the page specified in the `pageToOpen` property. Default: `true`
`pageToOpen`	`long`	The number of the page to import when opening a multipage document. Valid only when `importSinglePage` is `true`.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

OpenOptionsPhotoshop

Options for opening a Photoshop document, used with the open method.

OpenOptionsPhotoshop properties

Property	Value type	What it is
`layerComp`	`string`	The name of the layer comp to use when the document is converted.
`preserveHiddenLayers`	`boolean`	If `true`, preserve hidden layers when the document is converted. Default: `false`.
`preserveImageMaps`	`boolean`	If `true`, preserve image maps when the document is converted. Default: `true`.
`preserveLayers`	`boolean`	If `true`, preserve layers when the document is converted. Default: `true`.
`preserveSlices`	`boolean`	If `true`, preserve slices when the document is converted. Default: `true`.
`typename`	`string`	Read-only. The class name of the object.

PageItem

Any art item. Every art item and group in a document is a page item. You may refer to a page item as an element of a document, layer, or group item.

The `PageItem` class gives you complete access to every art item contained in an Illustrator document. The `PageItem` class is the superclass of all artwork objects in a document. The `CompoundPathItem`, `GroupItem`, `MeshItem`, `PathItem`, `PlacedItem`, `PluginItem`, `RasterItem`, and `TextFrame` classes each inherit a set of properties from the `PageItem` class.

You cannot create a `PageItem` directly, you must create one of the specific `PageItem` subclasses, such as `PathItem`.

PageItem properties

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout.
`blendingMode`	`BlendModes`	The mode to use when compositing this object. An object is considered composited when its opacity is set to less than 100.0 (100%).
`controlBounds`	`rect`	Read-only. The bounds of the object including stroke width and controls.
`editable`	`boolean`	Read-only. If `true`, this page item is editable.
`geometricBounds`	`rect`	Read-only. The object's bounds excluding the stroke width.
`height`	`real`	The height of the page item, calculated from the geometric bounds. Range: 0.0 to 16348.0
`hidden`	`boolean`	If `true`, this page item is hidden.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layer`	`Layer`	Read-only. The layer to which this page item belongs.
`left`	`number (double)`	The left position of the art item.
`locked`	`boolean`	If `true`, this page item is locked.
`name`	`string`	The name of this page item.
`note`	`string`	The note assigned to this item.
`opacity`	`real`	The opacity of this object, where 100.0 is completely opaque and 0.0 is completely transparent.
`parent`	object	Read-only. The parent of this object.
`pixelAligned`	`boolean`	True if this item is aligned to the pixel grid.
`position`	`point`	The position (in points) of the top left corner of the item in the format {x, y}. Does not include stroke weight.

Property	Value type	What it is
selected	boolean	If true, this object is selected.
sliced	boolean	If true, preserve slices.
tags	Tags	The collection of tags associated with this page item.
top	number (double)	The top position of the art item.
typename	string	Read-only. The class name of the object.
URL	string	The value of the Adobe URL tag assigned to this page item.
visibilityVariable	anything	The visibility variable to which this page item path is bound.
visibleBounds	rect	Read-only. The object's visible bounds, including stroke width of any objects in the illustration.
width	real	The width of the page item, calculated from the geometric bounds. Range: 0.0 to 16348.0
wrapInside	boolean	If true, the text frame object should be wrapped inside this object.
wrapOffset	number (double)	The offset to use when wrapping text around this object.
wrapped	boolean	If true, wrap text frame objects around this object (text frame must be above the object).
zOrderPosition	number (long)	Read-only. The drawing order of the art within its group or layer.

PageItem methods

Method	Parameter type	Returns	What it does
bringInPerspective (posX, posY, perspectiveGridPlane)	 number number PerspectiveGrid PlaneType	Nothing	Places art object(s) in a perspective grid at a specified position and grid plane.
resize (scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])	 number (double) number (double) boolean boolean boolean boolean number (double) Transformation	Nothing	Scales art object(s).

Method	Parameter type	Returns	What it does

`rotate`
`(angle`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,rotateAbout])`
	
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`Transformation`
		Nothing	Rotates art object(s).

`transform`
`(transformationMatrix`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidth]`
`[,transformAbout])`
	
	`matrix`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Transforms art object(s) using a transformation matrix.

`translate`
`([deltaX]`
`[,deltaY]`
`[,transformObjects]`
`[,transformFillPatterns]`
`[,transformFillGradents]`
`[,transformStrokePattern])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
		Nothing	Repositions art object(s).

`zOrder` `(zOrderCmd)`	 `ZOrderMethod`	Nothing	Arranges the art relative to other art in the group or layer.

PageItems

A collection of page item objects. Provides complete access to all the art items in an Illustrator document in the following classes:

CompoundPathItem
GraphItem
GroupItem
LegacyTextItem
MeshItem
NonNativeItem
PathItem
PlacedItem
PluginItem
RasterItem
SymbolItem
TextFrameItem

You can reference page items through the `PageItems` property in a `Document`, `Layer`, or `Group`. When you access an individual item in one of these collections, the reference is a page item of one of a particular type. For example, if you use `PageItems` to reference a graph item, the `typename` value of that object is `GraphItem`.

PageItems properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

PageItems methods

Method	Parameter type	Returns	What it does
`getByName (name)`	`string`	PageItem	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	PageItem	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Getting references to external files in page items

Before running this script, open a document that contains one or more linked images.

```
// Gets all file-references in the current document using the pageItems object,
// then displays them in a new document

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

    var sourceDoc = app.activeDocument;
    var sourceName =sourceDoc.name;
    for ( i = 0; i < sourceDoc.pageItems.length; i++ ) {
        artItem = sourceDoc.pageItems[i];
        switch ( artItem.typename ) {
            case "PlacedItem":
                fileReferences.push( artItem.file.fsName );
                break;
            case "RasterItem":
                if ( ! artItem.embedded ) {
                    fileReferences.push( artItem.file.fsName );
                }
                break;
        }
    }
    // Write the file references to a new document
    var reportDoc = documents.add();
    var areaTextPath = reportDoc.pathItems.rectangle( reportDoc.height,0,
         reportDoc.width, reportDoc.height );
    var fileNameText = reportDoc.textFrames.areaText( areaTextPath );
    fileNameText.textRange.size = 24;
    var paragraphCount = 3;
    var text = "File references in \'" + sourceName + "\':\r\r";
    for ( i = 0; i < fileReferences.length; i++ ) {
        text += ( fileReferences[i] + "\r" );
        paragraphCount++;
    }
    fileNameText.contents = text;
}
```

Paper

Associates paper information with a paper name. `Paper` objects are used by `Printer` objects.

Paper properties

Property	Value type	What it is
`name`	`string`	The paper name.
`paperInfo`	`PaperInfo`	The paper information.
`typename`	`string`	Read-only. The class name of the object.

PaperInfo

Paper information for use in printing documents.

PaperInfo properties

Property	Value type	What it is
`customPaper`	`boolean`	If `true`, it is a custom paper.
`height`	`number (double)`	The paper's height in points.
`imageableArea`	`array` of 4 numbers	The imageable area.
`typename`	`string`	Read-only. The class name of the object.
`width`	`number (double)`	The paper's width in points.

Finding paper information

```
// Displays the papers and paper sizes available for the 2nd printer in a text frame

var docRef = documents.add();
var itemRef = docRef.pathItems.rectangle(600, 300, 200, 100);
var textRef = docRef.textFrames.add();
textRef.top = 600;
textRef.left = 50;
// get paper objects for 2nd printer
var printerRef = printerList[1];
textRef.contents = printerRef.name;
textRef.contents += " paper list:\r";
var paragraphCount = 2;
// get details of each paper
var iCount = printerRef.printerInfo.paperSizes.length;
for( var i=0; i<iCount; i++ ) {
    var paperRef = printerRef.printerInfo.paperSizes[i];
    var paperInfoRef = paperRef.paperInfo;
    textRef.contents +=  paperRef.name;
    textRef.contents += "\t";
    textRef.contents += paperInfoRef.height;
    textRef.contents += " x ";
    textRef.contents += paperInfoRef.width;
    textRef.contents += "\r";
    paragraphCount++;
}
  redraw();
```

ParagraphAttributes

Specifies the properties and attributes of a paragraph contained in a text frame.

Note: Paragraph attributes do not have default values, and are undefined until explicitly set.

ParagraphAttributes properties

Property	Value type	What it is
`autoLeadingAmount`	`number (double)`	Auto leading amount expressed as a percentage.
`bunriKinshi`	`boolean`	If `true`, BunriKinshi is enabled.
`burasagariType`	`BurasagariTypeEnum`	The Burasagari type.
`desiredGlyphScaling`	`number (double)`	Desired glyph scaling, expressed as a percentage of the default character width. Range: 50.0 to 200.0. At 100.0, the width of characters is not changed.
`desiredLetterSpacing`	`number (double)`	Desired letter, spacing expressed as a percentage of the default kerning or tracking Range: -100.0 to 500.0. At 0, no space is added between letters. At 100.0, an entire space width is added between letters.
`desiredWordSpacing`	`number (double)`	Desired word spacing, expressed as a percentage of the default space for the font. Range: 0.0 to 1000.0; at 100.00. No space is added between words.
`everyLineComposer`	`boolean`	If `true`, the Every-line Composer is enabled. If `false`, the Single-line Composer is enabled.
`firstLineIndent`	`number (double)`	First line left indent in points.
`hyphenateCapitalizedWords`	`boolean`	If `true`, hyphenation is enabled for capitalized words.
`hyphenation`	`boolean`	If `true`, hyphenation is enabled for the paragraph.
`hyphenationPreference`	`number (double)`	Hyphenation preference scale for better spacing (0) or fewer hyphens (1). Range: 0.0 to 1.0

Property	Value type	What it is
hyphenationZone	number (double)	The distance (in points) from the right edge of the paragraph that marks the part of the line where hyphenation is not allowed. **NOTE:** 0 allows all hyphenation. Valid only when everyLineComposer is false.
justification	Justification	Paragraph justification.
kinsoku	string	The Kinsoku Shori name.
kinsokuOrder	KinsokuOrderEnum	The preferred Kinsoku order.
kurikaeshiMojiShori	boolean	If true, KurikaeshiMojiShori is enabled.
leadingType	AutoLeadingType	Auto leading type.
leftIndent	number (double)	The left indent of margin in points.
maximumConsecutiveHyphens	number (long)	Maximum number of consecutive hyphenated lines.
maximumGlyphScaling	number (double)	Maximum glyph scaling, expressed as a percentage of the default character width. Range: 50.0 to 200.0; at 100.0. The width of characters is not changed. **NOTE:** Valid only for justified paragraphs.
maximumLetterSpacing	number (double)	Maximum letter spacing, expressed as a percentage of the default kerning or tracking Range: -100.0 to 500.0; at 0. No space is added between letters. At 100.0, an entire space width is added between letters. **NOTE:** Valid only for justified paragraphs.
maximumWordSpacing	number (double)	Maximum word spacing, expressed as a percentage of the default space for the font. Range: 0.0 to 1000.0; at 100.00. No space is added between words. **NOTE:** Valid only for justified paragraphs.
minimumAfterHyphen	number (long)	Minimum number of characters after a hyphen.
minimumBeforeHyphen	number (long)	Minimum number of characters before a hyphen.

Property	Value type	What it is
`minimumGlyphScaling`	`number (double)`	Minimum glyph scaling, expressed as a percentage of the default character width. Range: 50.0 to 200.0. At 100.0, the width of characters is not changed. **NOTE:** Valid only for justified paragraphs.
`minimumHyphenatedWordSize`	`number (long)`	Minimum number of characters for a word to be hyphenated.
`minimumLetterSpacing`	`number (double)`	Minimum letter spacing, expressed as a percentage of the default kerning or tracking Range: -100.0 to 500.0; at 0. No space is added between letters. At 100.0, an entire space width is added between letters. **NOTE:** Valid only for justified paragraphs.
`minimumWordSpacing`	`number (double)`	Minimum word spacing, expressed as a percentage of the default space for the font. Range: 0.0 to 1000.0; at 100.00. No space is added between words. **NOTE:** Valid only for justified paragraphs.
`mojikumi`	`string`	The Mojikumi name.
`parent`	object	Read-only. The object's container.
`rightIndent`	`number (double)`	Right indent of margin in points.
`romanHanging`	`boolean`	If `true`, Roman hanging punctuation is enabled.
`singleWordJustification`	`Justification`	Single word justification.
`spaceAfter`	`number (double)`	Spacing after paragraph in points.
`spaceBefore`	`number (double)`	Spacing before paragraph in points.
`tabStops`	`TabStopInfo`	Tab stop settings.
`typename`	`string`	Read-only. The class name of the object.

Changing justification in paragraphs

```
// Creates a new document with 1 text frame and 3 paragraphs
// then gives each paragraph a different justification

var docRef = documents.add();
var pathRef = docRef.pathItems.rectangle(600, 200, 200, 400);
var textRef = docRef.textFrames.areaText(pathRef);
textRef.paragraphs.add("Left justified paragraph.");
textRef.paragraphs.add("Center justified paragraph.");
textRef.paragraphs.add("Right justified paragraph.");
textRef.textRange.characterAttributes.size = 28;

// change the justification of each paragraph
// using the paragraph attributes object
var paraAttr_0 = textRef.paragraphs[0].paragraphAttributes;
paraAttr_0.justification = Justification.RIGHT;
var paraAttr_1 = textRef.paragraphs[1].paragraphAttributes;
paraAttr_1.justification = Justification.CENTER;
var paraAttr_2 = textRef.paragraphs[2].paragraphAttributes;
paraAttr_2.justification = Justification.LEFT;
```

Paragraphs

A collection of `TextRange` objects, with each `TextRange` representing a paragraph. The elements are not named; you must access them by index.

Paragraphs properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

Paragraphs methods

Method	Parameter type	Returns	What it does
`add (contents [,relativeObject] [,insertionLocation])`	`string` `TextFrameItem` `ElementPlacement`	`TextRange`	Adds a new paragraph with specified text contents at the specified location in the current document. If location is not specified, adds the new paragraph to the containing text frame after the current text selection or insertion point.
`addBefore (contents)`	`string`	`TextRange`	Adds a new paragraph with specified text contents before the current text selection or insertion point.
`index (itemKey)`	`number`	`TextRange`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Counting paragraphs

```
// Counts all paragraphs in current doc and stores result in paragraphCount

if ( app.documents.length > 0 ) {
    doc = app.activeDocument;
    paragraphCount = 0;
    for ( i = 0; i < doc.textFrames.length; i++ ) {
        paragraphCount += doc.textFrames[i].paragraphs.length;
    }
}
```

ParagraphStyle

Associates character and paragraph attributes with a style name. The style object can be used to apply those attributes to the text in a `TextFrame` object. See Creating and applying a paragraph style below.

ParagraphStyle properties

Property	Value type	What it is
`characterAttributes`	`CharacterAttributes`	Read-only. The character properties for the text range.
`name`	`string`	The paragraph style's name.
`paragraphAttributes`	`ParagraphAttributes`	Read-only. The paragraph properties for the text range.
`parent`	object	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the object.

ParagraphStyle methods

Method	Parameter type	Returns	What it does
`applyTo (textItem [,clearingOverrides])`	object `boolean`	Nothing	Applies this paragraph style to the specified text item.
`remove ()`		Nothing	Deletes the object.

ParagraphStyles

A collection of `ParagraphStyle` objects.

ParagraphStyles properties

Property	Value type	What it is
`length`	`number`	Read-only. Number of elements in the collection.
`parent`	object	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the object.

ParagraphStyles methods

Method	Parameter type	Returns	What it does
`add (name)`	`string`	`ParagraphStyle`	Creates a named paragraph style.
`getByName (name)`	`string`	`ParagraphStyle`	Get the first element in the collection with the provided name.
`index (itemKey)`	`string, number`	`ParagraphStyle`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in the collection.

Creating and applying a paragraph style

```
// Creates a new document with 1 text frame and 3 paragraphs
// gives each paragraph a different justification, then creates
// a paragraph style and applies it to all paragraphs

var docRef = documents.add();
var pathRef = docRef.pathItems.rectangle(600, 200, 200, 400);
var textRef = docRef.textFrames.areaText(pathRef);
textRef.paragraphs.add("Left justified paragraph.");
textRef.paragraphs.add("Center justified paragraph.");
textRef.paragraphs.add("Right justified paragraph.");
textRef.textRange.characterAttributes.size = 28;

// change the justification of each paragraph
// using the paragraph attributes object
var paraAttr_0 = textRef.paragraphs[0].paragraphAttributes;
paraAttr_0.justification = Justification.RIGHT;
var paraAttr_1 = textRef.paragraphs[1].paragraphAttributes;
paraAttr_1.justification = Justification.CENTER;
var paraAttr_2 = textRef.paragraphs[2].paragraphAttributes;
paraAttr_2.justification = Justification.LEFT;

// create a new paragraph style
var paraStyle = docRef.paragraphStyles.add("LeftIndent");

// add some paragraph attributes
var paraAttr = paraStyle.paragraphAttributes;
paraAttr.justification = Justification.LEFT;
paraAttr.firstLineIndent = 10;

// apply the style to each item in the document
var iCount = textRef.paragraphs.length;
for(var i=0; i<iCount; i++) {
   paraStyle.applyTo(textRef.paragraphs[i], true);
}
redraw();
```

PathItem

Specifies a path item, which contains `PathPoint` objects that define its geometry. The `PathItem` class gives you complete access to paths in Illustrator. The `setEntirePath` method provides an extremely efficient way to create paths comprised of straight lines.

PathItem properties

Property	Value type	What it is
`area`	`number (double)`	Read-only. The area of this path in square points. If the area is negative, the path is wound counterclockwise. Self-intersecting paths can contain sub-areas that cancel each other out, which makes this value zero even though the path has apparent area.
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout.
`blendingMode`	`BlendModes`	The blend mode used when compositing an object.
`clipping`	`boolean`	If `true`, this path should be used as a clipping path.
`closed`	`boolean`	If `true`, this path is closed.
`controlBounds`	`array` of 4 numbers	Read-only. The bounds of the object including stroke width and controls.
`editable`	`boolean`	Read-only. If `true`, this item is editable.
`evenodd`	`boolean`	If `true`, the even-odd rule should be used to determine "insideness."
`fillColor`	`Color`	The fill color of the path.
`filled`	`boolean`	If `true`, the path be filled.
`fillOverprint`	`boolean`	If `true`, the art beneath a filled object should be overprinted.
`geometricBounds`	`array` of 4 numbers	Read-only. The bounds of the object excluding stroke width.
`guides`	`boolean`	If `true`, this path is a guide object.
`height`	`number (double)`	The height of the group item.
`hidden`	`boolean`	If `true`, this item is hidden.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layer`	`Layer`	Read-only. The layer to which this item belongs.
`left`	`number (double)`	The position of the left side of the item (in points, measured from the left side of the page).

Property	Value type	What it is
length	number (double)	The length of this path in points.
locked	boolean	If true, this item is locked.
name	string	The name of this item.
note	string	The note text assigned to the path.
opacity	number (double)	The opacity of the object. Range: 0.0 to 100.0
parent	Layer or GroupItem	Read-only. The parent of this object.
pathPoints	PathPoints	Read-only. The path points contained in this path item.
pixelAligned	boolean	True if this item is aligned to the pixel grid.
polarity	PolarityValues	The polarity of the path.
position	array of 2 numbers	The position (in points) of the top left corner of the pathItem object in the format [x, y]. Does not include stroke weight.
resolution	number (double)	The resolution of the path in dots per inch (dpi).
selected	boolean	If true, this item is selected.
selectedPathPoints	PathPoints	Read-only. All of the selected path points in the path.
sliced	boolean	If true, the item sliced. Default: false
strokeCap	StrokeCap	The type of line capping.
strokeColor	Color	The stroke color for the path.
stroked	boolean	If true, the path should be stroked.
strokeDashes	object	Dash lengths. Set to an empty object, {}, for a solid line.
strokeDashOffset	number (double)	The default distance into the dash pattern at which the pattern should be started.
strokeJoin	StrokeJoin	Type of joints for the path.
strokeMiterLimit	number (double)	When a default stroke join is set to mitered, this property specifies when the join will be converted to beveled (squared-off) by default. The default miter limit of 4 means that when the length of the point reaches four times the stroke weight, the join switches from a miter join to a bevel join. A value of 1 specifies a bevel join. Range: 1 to 500. Default: 4
strokeOverprint	boolean	If true, the art beneath a stroked object should be overprinted.
strokeWidth	number (double)	The width of the stroke (in points).

Property	Value type	What it is
tags	Tags	Read-only. The tags contained in this item.
top	number (double)	The position of the top of the item (in points, measured from the bottom of the page).
typename	string	Read-only. The class name of the referenced object.
uRL	string	The value of the Adobe URL tag assigned to this item.
visibilityVariable	Variable	The visibility variable bound to the item.
visibleBounds	array of 4 numbers	Read-only. The visible bounds of the item including stroke width.
width	number (double)	The width of the item.
wrapInside	boolean	If true, the text frame object should be wrapped inside this object.
wrapOffset	number (double)	The offset to use when wrapping text around this object.
wrapped	boolean	If true, wrap text frame objects around this object (text frame must be above the object).
zOrderPosition	number (long)	Read-only. The position of this item within the stacking order of the group or layer (parent) that contains the item.

PathItem methods

Method	Parameter type	Returns	What it does
duplicate ([relativeObject] [,insertionLocation])	object ElementPlacement	PathItem	Creates a duplicate of the selected object.
move (relativeObject, insertionLocation)	object ElementPlacement	PathItem	Moves the object.
remove ()		Nothing	Deletes this object.
resize (scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])	number (double) number (double) boolean boolean boolean boolean number (double) Transformation	Nothing	Scales the art item 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 (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`Transformation`
		Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the `angle` value is positive, clockwise if the value is negative.

`setEntirePath` `(pathPoints)`	`array` of `[x, y]` coordinate pairs	Nothing	Sets the path using an array of points specified as [x, y] coordinate pairs.

`transform`
`(transformationMatrix`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,transformAbout])`
	
	`Matrix`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Transforms the art item by applying a transformation matrix.

`translate`
`([deltaX]`
`[,deltaY]`
`[,transformObjects]`
`[,transformFillPatterns]`
`[,transformFillGradients]`
`[,transformStrokePatterns])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
		Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.

`zOrder` `(zOrderCmd)`	 `ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

Setting colors in a path

```
// Sets the stroke and fill of a path item to colors of a randomly selected swatch

if ( app.documents.length > 0 && app.activeDocument.pathItems.length > 0 ) {
    doc = app.activeDocument;
    for (var i = 0; i < doc.pathItems.length; i++ ) {
        pathRef = doc.pathItems[i];
        pathRef.filled = true;
        pathRef.stroked = true;
        swatchIndex = Math.round( Math.random() * ( doc.swatches.length - 1 ) );
        pathRef.fillColor = doc.swatches[ swatchIndex ].color;
        pathRef.strokeColor = doc.swatches[ swatchIndex ].color;
    }
}
```

Creating a path from straight lines

This script illustrates the use of the `setEntirePath` method.

```
// Creates a new open path consisting of 10 straight lines

if ( app.documents.length > 0 ) {
    var lineList = new Array(10);
    for ( i = 0; i < lineList.length; i++ ) {
        lineList[i] = new Array( i * 10 + 50, ((i - 5) ^ 2) * 5 +50);
    }
    app.defaultStroked = true;
    newPath = app.activeDocument.pathItems.add();
    newPath.setEntirePath(lineList);
}
```

PathItems

A collection of `PathItem` objects. 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 are used.

PathItems properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

PathItems methods

Method	Parameter type	Returns	What it does
`add ()`		`PathItem`	Creates a new object.
`ellipse ([top] [,left] [,width] [,height] [,reversed] [,inscribed])`	`number (double)` `number (double)` `number (double)` `number (double)` `boolean` `boolean`	`PathItem`	Creates a new pathItem in the shape of an ellipse using the supplied parameters. Defaults: `top`: 100 pt.; `left`: 100 pt.; `width`: 50 pt.; `height`: 100 pt.; `reversed`: `false`
`getByName (name)`	`string`	`PathItem`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`PathItem`	Gets an element from the collection.
`polygon ([centerX] [,centerY] [,radius] [,sides] [,reversed])`	`number (double)` `number (double)` `number (double)` `number (long)` `boolean`	`PathItem`	Creates a new pathItem in the shape of an polygon using the supplied parameters. Defaults: `centerX`: 200 pt.; `centerY`: 300 pt.; `radius`: `50` pt.; `sides`: 8; `reversed`: `false`
`rectangle (top, left, width, height [,reversed])`	`number (double)` `number (double)` `number (double)` `number (double)` `boolean`	`PathItem`	Creates a new pathItem in the shape of an polygon using the supplied parameters.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Method	Parameter type	Returns	What it does
roundedRectangle (top, left, width, height [,horizontalRadius] [,verticalRadius] [,reversed])	 number (double) number (double) number (double) number (double) number (double) number (double) boolean	PathItem	Creates a new pathItem in the shape of a rectangle with rounded corners using the supplied parameters. Defaults: horizontalRadius: 15 pt.; verticalRadius: 20 pt.; reversed: false
star ([centerX] [,centerY] [,radius] [,innerRadius] [,points] [,reversed])	 number (double) number (double) number (double) number (double) number (long) boolean	PathItem	Creates a new path item in the shape of a star using the supplied parameters. Defaults: centerX: 200 pt.; centerY: 300 pt.; radius: 50 pt.; innerRadius: 20 pt.; points: 5; reversed: false

Creating shapes

```
// Creates 5 shapes in layer 1 of document 1
// and applies a random graphic style to each

var doc = app.documents.add();
var artLayer = doc.layers[0];
app.defaultStroked = true;
app.defaultFilled = true;

var rect = artLayer.pathItems.rectangle( 762.5, 87.5, 425.0, 75.0 );
var rndRect = artLayer.pathItems.roundedRectangle(
          637.5, 87.5, 425.0, 75.0, 20.0, 10.0 );
// Create ellipse, 'reversed' is false, 'inscribed' is true
var ellipse = artLayer.pathItems.ellipse(
          512.5, 87.5, 425.0, 75.0, false, true );
// Create octagon, and 8-sided polygon
var octagon = artLayer.pathItems.polygon( 300.0, 325.0, 75.0, 8 );
// Create a 4 pointed star
var star = artLayer.pathItems.star( 300.0, 125.0, 100.0, 20.0, 4 );

for ( i = 0; i < artLayer.pathItems.length; i++ ) {
    styleIndex = Math.round(
              Math.random() * ( doc.graphicStyles.length - 1 ) );
    doc.graphicStyles[styleIndex].applyTo( artLayer.pathItems[i] );
}
```

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`).

PathPoint properties

Property	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`	`PathItem`	Read-only. The path item that contains this path point.
`pointType`	`PointType`	The type of path point, either a curve or a corner. Any point can considered a corner point. Setting the type 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.
`rightDirection`	`array` of 2 numbers	The position of this path point's out control point.
`selected`	`PathPointSelection`	Are points of this path point selected, and if so, which ones.
`typename`	`string`	Read-only. The class name of the referenced object.

PathPoint methods

Method	Parameter type	Returns	What it does
`remove ()`		Nothing	Removes the referenced point from the path.

PathPoints

A collection of `PathPoint` objects in a specific path. The elements are not named; you must access them by index.

PathPoints properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

PathPoints methods

Method	Parameter type	Returns	What it does
`add ()`		`PathPoint`	Creates a new `PathPoint` object.
`index (itemKey)`	`number`	`PathPoint`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Adding a path point to a path

```
// Appends a new PathPoint to an existing path
// and initializes its anchor and handle points.

if ( app.documents.length > 0 ) {
    var doc = app.activeDocument;
    var line = doc.pathItems.add();
    line.stroked = true;
    line.setEntirePath( Array( Array(220, 475), Array(375, 300) ) );

    // Append another point to the line
    var newPoint = doc.pathItems[0].pathPoints.add();

    newPoint.anchor = Array(220, 300);
    newPoint.leftDirection = newPoint.anchor;
    newPoint.rightDirection = newPoint.anchor;
    newPoint.pointType = PointType.CORNER;
}
```

Pattern

An Illustrator pattern definition contained in a document. Patterns are shown in the Swatches palette. Each pattern is referenced by a PatternColor object, which defines the pattern's appearance.

Pattern properties

Property	Value type	What it is
`name`	`string`	The pattern name.
`parent`	`Document`	Read-only. The document that contains this pattern.
`typename`	`string`	Read-only. The class name of the referenced object.

Pattern methods

Method	Parameter type	Returns	What it does
`remove ()`		Nothing	Removes the referenced pattern from the document.
`toString ()`		`string`	Returns the object type of a referenced object. If the object has a name, also returns the name.

PatternColor

A pattern color specification. You can create a new pattern color by modifying an existing pattern in the document. Any modification you make to a pattern affects that pattern in the Palette.

`PatternColor` objects can be used in any property that takes a color object, such as `fillColor` or `strokeColor`.

PatternColor properties

Property	Value type	What it is
`matrix`	`Matrix`	Additional transformation arising from manipulating the path.
`pattern`	`Pattern`	A reference to the pattern object that defines the pattern to use in this color definition.
`reflect`	`boolean`	If `true`, the prototype should be reflected before filling. Default: `false`
`reflectAngle`	`number (double)`	The axis around which to reflect, in points. Default: 0.0
`rotation`	`number (double)`	The angle in radians to rotate the prototype pattern before filling. Default: 0.0
`scaleFactor`	`array` of 2 numbers	The fraction to which to scale the prototype pattern before filling, represented as a point containing horizontal and vertical scaling percentages.
`shearAngle`	`number (double)`	The angle in radians by which to slant the shear. Default: 0.0
`shearAxis`	`number (double)`	The axis to shear with respect to, in points. Default: 0.0
`shiftAngle`	`number (double)`	The angle in radians to which to translate the unscaled prototype pattern before filling. Default: 0.0
`shiftDistance`	`number (double)`	The distance in points to which to translate the unscaled prototype pattern before filling. Default: 0.0
`typename`	`string`	Read-only. The class name of the referenced object.

Modifying and applying pattern colors

```
// Rotates the color of each pattern in the current document,
// then applies the last pattern to the first path item

if ( app.documents.length > 0 && app.activeDocument.pathItems.length > 0 ) {
    doc = app.activeDocument;
    swatchIndex = 0;
    for ( i = 0; i < doc.swatches.length; i++ ) {
        // Get the generic color object of the swatch
        currentSwatch = doc.swatches[i];
        swatchColor = currentSwatch.color;
        // Only operate on patterns
        if ( currentSwatch.color.typename == "PatternColor" ) {
            // Change a pattern property
            currentSwatch.color.rotation = 10;
            swatchIndex = i;
        }
    }
// Apply the last pattern color swatch to the frontmost path
    firstPath = app.activeDocument.pathItems[0];
    firstPath.filled = true;
    firstPath.fillColor = doc.swatches[swatchIndex].color;
}
```

Patterns

A collection of `Pattern` objects in a document.

Patterns properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

Patterns methods

Method	Parameter type	Returns	What it does
`add ()`		`Pattern`	Creates a new object.
`getByName (name)`	`string`	`Pattern`	Gets the first element in the collection with the provided name.
`index (itemKey)`	`string, number`	`Pattern`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Removing a pattern

```
// Deletes the last pattern from the current document.

if ( app.documents.length > 0 ) {
    var lastIndex = app.activeDocument.patterns.length - 1;
    var patternToRemove = app.activeDocument.patterns[lastIndex];
    var patternName = patternToRemove.name;
    patternToRemove.remove();
    // Note after removing Illustrator objects, set the variable that
    // referenced the removed object to null, since it is now invalid.
    patternToRemove = null;
}
```

PDFFileOptions

Options for opening a PDF file, used with the open method. All properties are optional.

PDFFileOptions properties

Property	Value type	What it is
`pageToOpen`	`number (long)`	What page should be used when opening a multipage document. Default: 1
`parent`	object	Read-only. The object's container.
`pDFCropToBox`	`PDFBoxType`	Which box should be used when placing a multipage document. Default: `PDFBoxType.PDFMediaBox`
`typename`	`string`	Read-only. The class name of the object.

Opening a PDF with options

```
// Opens a PDF file with specified options

var pdfOptions = app.preferences.PDFFileOptions;
pdfOptions.pDFCropToBox = PDFBoxType.PDFBOUNDINGBOX;
pdfOptions.pageToOpen = 2;

// Open a file using these preferences
var fileRef = filePath;
if (fileRef != null) {
  var docRef = open(fileRef, DocumentColorSpace.RGB);
}
```

PDFSaveOptions

Options for saving a document as an Adobe PDF file, used with the saveAs method. All properties are optional.

PDFSaveOptions properties

Property	Value type	What it is
`acrobatLayers`	`boolean`	Optional. Create Acrobat® layers from top-level layers. Acrobat 6 only. Default: `false`
`artboardRange`	`string`	Optional. This is considered for multi-asset extraction, which specifies the artboard range. An empty string extracts all the artboards. Default: empty string
`bleedLink`	`boolean`	Optional. Link 4 bleed values. Default: `true`
`bleedOffsetRect`	`array` of 4 numbers	The bleed offset rectangle.
`colorBars`	`boolean`	Optional. Draw color bars. Default: `false`
`colorCompression`	`CompressionQuality`	Optional. The type of color bitmap compression used. Default: `CompressionQuality.None`
`colorConversionID`	`ColorConversion`	Optional. The PDF color conversion policy. Default: `ColorConversion.None`
`colorDestinationID`	`ColorDestination`	Optional. The conversion target for color conversion. Default: `ColorDestination.None`
`colorDownsampling`	`number (double)`	Optional. The color downsampling resolution in dots per inch (dpi). If 0, no downsampling is performed. Default: 150.0
`colorDownsamplingImageThreshold`	`number (double)`	Optional. Downsample if the image's resolution is above this value. Default: 225.0

Property	Value type	What it is
colorDownsamplingMethod	DownsampleMethod	Optional. How color bitmap images should be resampled. Default: DownsampleMethod.NODOWNSAMPLE
colorProfileID	ColorProfile	Optional. The color profile to include. Default: ColorProfile.None
colorTileSize	number (long)	Optional. Tile size when compressing with JPEG2000. Default: 256
compatibility	PDFCompatibility	Optional. The version of the Acrobat file format to create. Default: PDFCompatibility.Acrobat5
compressArt	boolean	Optional. If true, the line art and text should be compressed. Default: true
documentPassword	string	Optional. A password string to open the document. Default: no string
enableAccess	boolean	Optional. If true, enable accessing 128-bit. Default: true
enableCopy	boolean	Optional. If true, enable copying of text 128-bit. Default: true
enableCopyAccess	boolean	Optional. If true, enable copying and accessing 40-bit. Default: true
enablePlainText	boolean	Optional. If true, enable plaintext metadata 128-bit. Available only for Acrobat 6. Default: false
flattenerOptions	PrintFlattenerOptions	Optional. The printing flattener options.
flattenerPreset	stringOptional.	Optional. The transparency flattener preset name.

Property	Value type	What it is
`fontSubsetThreshold`	`number (double)`	Optional. Include a subset of fonts when less than this percentage of characters is used in the document. Valid for Illustrator 9 file format. Range: 0.0 to 100.0. Default: 100.0
`generateThumbnails`	`boolean`	Optional. If `true`, thumbnail images are generated with the saved file. Default: `true`
`grayscaleCompression`	`CompressionQuality`	Optional. Quality of grayscale bitmap compression. Default: `None`
`grayscaleDownsampling`	`number (double)`	Optional. Downsampling resolution in dots per inch (dpi). If 0, no downsampling is performed. Default: 150.0
`grayscaleDownsamplingImageThreshold`	`number (double)`	Optional. Downsample if the image's resolution is above this value. Default: 225.0
`grayscaleDownsamplingMethod`	`DownsampleMethod`	Optional. How grayscale bitmap images should be resampled Default: `DownSampleMethod.NODOWNSAMPLE`
`grayscaleTileSize`	`number (long)`	Optional. Tile size when compressing with JPEG2000. Default: 256
`monochromeCompression`	`MonochromeCompression`	Optional. Type of monochrome bitmap compression used. Default: `MonochromeCompression.None`
`monochromeDownsampling`	`number (double)`	Optional. Downsampling resolution in dots per inch (dpi). If 0, no downsampling is performed. Default: 300
`monochromeDownsamplingImageThreshold`	`number (double)`	Optional. Downsample if the image's resolution is above this value. Default: 450.0

Property	Value type	What it is
monochromeDownsamplingMethod	DownsampleMethod	Optional. How monochrome bitmap images should be resampled. Default: DownSampleMethod.NODOWNSAMPLE
offset	number (double)	Optional. Custom offset in points for using the custom paper. Default: 0.0
optimization	boolean	Optional. If true, the PDF document should be optimized for fast web viewing. Default: false
outputCondition	string	Optional. An optional comment to add to the PDF file, describing the intended printing condition. Default: not included
outputConditionID	string	Optional. The name of a registered printing condition. Default: not included
pageInformation	boolean	Optional. If true, raw page information. Default: false
pageMarksType	PageMarksTypes	Optional. The page marks style. Default: PageMarksType.Roman
pDFAllowPrinting	PDFPrintAllowedEnum	Optional. PDF security printing permission. Default: PDFPrintAllowedEnum.PRINT128HIGHRESOLUTION
pDFChangesAllowed	PDFChangesAllowedEnum	Optional. Security changes allowed. Default: PDFChangeAllowedEnum.CHANGE128ANYCHANGES
pDFPreset	string	Optional. Name of PDF preset to use.
pDFXStandard	PDFXStandard	Optional. The PDF standard with which this document complies. Default: PDFXStandard.PDFXNONE
pDFXStandardDescription	string	Optional. A description of the PDF standard from the selected preset.

Property	Value type	What it is
`permissionPassword`	`string`	Optional. A password string to restrict editing security settings. Default: no string
`preserveEditability`	`boolean`	Optional. If `true`, Illustrator editing capabilities should be preserved when saving the document. Default: `true`
`printerResolution`	`number (double)`	Optional. Flattening printer resolution. Default: 800.0
`registrationMarks`	`boolean`	Optional. If `true`, draw registration marks. Default: `false`
`requireDocumentPassword`	`boolean`	Optional. Require a password to open the document. Default: `false`
`requirePermissionPassword`	`boolean`	Optional. Use a password to restrict editing security settings. Default: `false`
`trapped`	`boolean`	Optional. If `true`, manual trapping has been prepared for the document. Default: `false`
`trimMarks`	`boolean`	Optional. Draw trim marks. Default: `false`
`trimMarkWeight`	`PDFTrimMarkWeight`	Optional. The trim mark weight. Default: `PDFTrimMarkWeight.TRIMMARKWEIGHT0125`
`typename`	`string`	Optional. Read-only. The class name of the referenced object.
`viewAfterSaving`	`boolean`	Optional. View PDF after saving. Default: `false`

Saving to PDF format

```
// Saves the current document as PDF to dest with specified options
// dest contains the full path and file name to save to

function saveFileToPDF (dest) {
    var doc = app.activeDocument;
    if ( app.documents.length > 0 ) {
        var saveName = new File ( dest );
        saveOpts = new PDFSaveOptions();
        saveOpts.compatibility = PDFCompatibility.ACROBAT5;
        saveOpts.generateThumbnails = true;
        saveOpts.preserveEditability = true;
        doc.saveAs( saveName, saveOpts );
    }
}
```

PhotoshopFileOptions

Options for opening a Photoshop file, used with the open method. All properties are optional.

PhotoshopFileOptions properties

Property	Value type	What it is
`parent`	object	Read-only. The parent of this object.
`pixelAspectRatioCorrection`	`boolean`	If `true`, imported images that have a non-square pixel aspect ratio should be adjusted.
`preserveImageMaps`	`boolean`	If `true`, image maps should be preserved when document is converted. Default: `true`
`preserveLayers`	`boolean`	If `true`, layers should be preserved when document is converted. Default: `true`
`preserveSlices`	`boolean`	If `true`, slices should be preserved when document is converted. Default: `true`
`typename`	`string`	Read-only. The class name of the referenced object.

Opening a Photoshop file

```
// Opens a Photoshop file containing layers with
// preferences set to preserve layers

var psdOptions = preferences.photoshopFileOptions;
psdOptions.preserveLayers = true;
psdOptions.pixelAspectRatioCorrection = false;
// open a file using these prefs
var fileRef = File( psdFilePath);
if (fileRef != null) {
  var docRef = open(fileRef, DocumentColorSpace.RGB);
}
```

PlacedItem

An artwork item placed in a document as a linked file. For example, an artwork object created using the File > Place command in Illustrator or using the `add()` method of the `placedItems` collection object is a placed item. For information, see "PlacedItems" on page 155.

PlacedItem properties

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout.
`blendingMode`	`BlendModes`	The blend mode used when compositing an object.
`boundingBox`	`array` of 4 numbers	Read-only. The dimensions of the placed art item regardless of transformations.
`contentVariable`	`Variable`	The content variable bound to the item.
`controlBounds`	`array` of 4 numbers	Read-only. The bounds of the object including stroke width and controls.
`editable`	`boolean`	Read-only. If `true`, this item is editable.
`file`	`File`	The file containing the artwork.
`geometricBounds`	`array` of 4 numbers	Read-only. The bounds of the object excluding stroke width.
`height`	`number (double)`	The height of the group item.
`hidden`	`boolean`	If `true`, this item is hidden.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layer`	`Layer`	Read-only. The layer to which this item belongs.
`left`	`number (double)`	The position of the left side of the item (in points, measured from the left side of the page).
`locked`	`boolean`	If `true`, this item is locked.
`matrix`	`Matrix`	The transformation matrix of the placed artwork.
`name`	`string`	The name of this item.
`note`	`string`	The note assigned to this item.
`opacity`	`number (double)`	The opacity of the object. Range: 0.0 to 100.0
`parent`	`Layer` or `GroupItem`	Read-only. The parent of this object.
`position`	`array` of 2 numbers	The position (in points) of the top left corner of the `placedItem` object in the format [x, y]. Does not include stroke weight.
`selected`	`boolean`	If `true`, this item is selected.

Property	Value type	What it is
sliced	boolean	If true, the item sliced. Default: false
tags	Tags	Read-only. The tags contained in this item.
top	number (double)	The position of the top of the item (in points, measured from the bottom of the page).
typename	string	Read-only. The class name of the referenced object.
uRL	string	The value of the Adobe URL tag assigned to this item.
visibilityVariable	Variable	The visibility variable bound to the item.
visibleBounds	array of 4 numbers	Read-only. The visible bounds of the item including stroke width.
width	number (double)	The width of the item.
wrapInside	boolean	If true, the text frame object should be wrapped inside this object.
wrapOffset	number (double)	The offset to use when wrapping text around this object.
wrapped	boolean	If true, wrap text frame objects around this object (text frame must be above the object).
zOrderPosition	number (long)	Read-only. The position of this item within the stacking order of the group or layer (parent) that contains the item.

PlacedItem methods

Method	Parameter type	Returns	What it does
duplicate ([relativeObject] [,insertionLocation])	object ElementPlacement	PlacedItem	Creates a duplicate of the selected object.
embed ()		Nothing	Embeds this art in the document. Converts the art to art item objects as needed and deletes this object.
move (relativeObject, insertionLocation)	object ElementPlacement	PlacedItem	Moves the object.
relink (linkFile)	File object	Nothing	Relinks the art object with the file that defines its content.
remove ()		Nothing	Deletes this object.

Method	Parameter type	Returns	What it does

`resize`
`(scaleX,`
`scaleY`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,scaleAbout])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Scales the art item 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 (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`Transformation`
		Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the `angle` value is positive, clockwise if the value is negative.

`trace` `()`		`PluginItem`	Converts the raster art for this object to vector art, using default options. Reorders the placed art into the source art of a plug-in group, and converts it into a group of filled and/or stroked paths that resemble the original image. Creates and returns a `pluginItem` object that references a `tracingObject` object.

`transform`
`(transformationMatrix`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,transformAbout])`
	
	`Matrix`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Transforms the art item by applying a transformation matrix.

`translate`
`([deltaX]`
`[,deltaY]`
`[,transformObjects]`
`[,transformFillPatterns]`
`[,transformFillGradients]`
`[,transformStrokePatterns])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
		Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.

`zOrder` `(zOrderCmd)`	 `ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

Changing the selection state of placed items

```
// Toggles the selection state of all placed items.

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

PlacedItems

A collection of `PlacedItem` objects in the document.

PlacedItems properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

PlacedItems methods

Method	Parameter type	Returns	What it does
`add ()`	`none`	`PlacedItem`	Creates a new object. Use to place new art in a document. Use the `file` property of the resulting `placedItem` object to link the file containing the artwork. See "PlacedItem" on page 151.
`getByName (name)`	`string`	`PlacedItem`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`PlacedItem`	Gets an element from the collection.
`removeAll ()`	`none`	Nothing	Deletes all elements in this collection.

PluginItem

An art item created by an Illustrator plug-in. Scripts can create a plug-in item using `PlacedItem.trace` or `RasterItem.trace`, and can copy existing plug-in items using the `duplicate` method, but cannot create `PluginItem` objects directly.

PluginItem properties

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout.
`blendingMode`	`BlendModes`	The blend mode used when compositing an object.
`controlBounds`	`array` of 4 numbers	Read-only. The bounds of the object including stroke width and controls.
`editable`	`boolean`	Read-only. If `true`, this item is editable.
`geometricBounds`	`array` of 4 numbers	Read-only. The bounds of the object excluding stroke width.
`height`	`number (double)`	The height of the group item.
`hidden`	`boolean`	If `true`, this item is hidden.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`isTracing`	`boolean`	If `true`, this plug-in group represents a vector art item created by tracing a raster art item. The `tracing` property contains the tracing object associated with the options used to create it.
`layer`	`Layer`	Read-only. The layer to which this item belongs.
`left`	`number (double)`	The position of the left side of the item (in points, measured from the left side of the page).
`locked`	`boolean`	If `true`, this item is locked.
`name`	`string`	The name of this item.
`note`	`string`	The note assigned to this item.
`opacity`	`number (double)`	The opacity of the object. Range: 0.0 to 100.0
`parent`	`Layer` or `GroupItem`	Read-only. The parent of this object.
`position`	`array` of 2 numbers	The position (in points) of the top left corner of the `plugInItem` object in the format [x, y]. Does not include stroke weight.
`selected`	`boolean`	If `true`, this item is selected.
`sliced`	`boolean`	If `true`, the item sliced. Default: `false`

Property	Value type	What it is
tags	Tags	Read-only. The tags contained in this item.
top	number (double)	The position of the top of the item (in points, measured from the bottom of the page).
tracing	TracingObject	When this plug-in group was created by tracing (isTracing is true), the tracing object associated with the options used to create it.
typename	string	Read-only. The class name of the referenced object.
uRL	string	The value of the Adobe URL tag assigned to this item.
visibilityVariable	Variable	The visibility variable bound to the item.
visibleBounds	array of 4 numbers	Read-only. The visible bounds of the item including stroke width.
width	number (double)	The width of the item.
wrapInside	boolean	If true, the text frame object should be wrapped inside this object.
wrapOffset	number (double)	The offset to use when wrapping text around this object.
wrapped	boolean	If true, wrap text frame objects around this object (text frame must be above the object).
zOrderPosition	number	Read-only. The position of this item within the stacking order of the group or layer (parent) that contains the item.

PluginItem methods

Method	Parameter type	Returns	What it does
duplicate ([relativeObject] [,insertionLocation])	object ElementPlacement	PluginItem	Creates a duplicate of the selected object.
move (relativeObject, insertionLocation)	object ElementPlacement	PluginItem	Moves the object.
remove ()		Nothing	Deletes this object.

Method	Parameter type	Returns	What it does

`resize`
`(scaleX,`
`scaleY`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,scaleAbout])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Scales the art item 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 (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`Transformation`
		Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the `angle` value is positive, clockwise if the value is negative.

`transform`
`(transformationMatrix`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,transformAbout])`
	
	`Matrix`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Transforms the art item by applying a transformation matrix.

`translate`
`([deltaX]`
`[,deltaY]`
`[,transformObjects]`
`[,transformFillPatterns]`
`[,transformFillGradients]`
`[,transformStrokePatterns])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
		Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.

`zOrder` `(zOrderCmd)`	 `ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

Copying a plug-in item

```
// Creates new plug-in art by copying an existing plug-in art item

if ( app.documents.length > 0 && app.activeDocument.pluginItems.length > 0 ) {
    doc = app.activeDocument;
    pluginArt = doc.pluginItems[0];
    pluginArt.duplicate( pluginArt.parent,
        ElementPlacement.PLACEATBEGINNING );
}
```

PluginItems

A collection of `PluginItem` objects in a document. See Copying a plug-in item.

PluginItems properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

PluginItems methods

Method	Parameter type	Returns	What it does
`getByName (name)`	`string`	`PluginItem`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`PluginItem`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all objects in this collection.

PPDFile

Associates file information with a PostScript Printer Description (PPD) file.

PPDFile properties

Property	Value type	What it is
`name`	`string`	The PPD model name.
`PPDInfo`	`PPDFileInfo`	The PPD file information.
`typename`	`string`	Read-only. The class name of the object.

PPDFileInfo

Information about a PostScript Printer Description (PPD) file.

PPDFileInfo properties

Property	Value type	What it is
`languageLevel`	`string`	The PostScript language level.
`PPDFilePath`	`File`	Path specification for the PPD file.
`screenList`	`array` of `Screen`	List of color separation screens.
`screenSpotFunctionList`	`array` of `ScreenSpotFunction`	List of color separation screen spot functions.

Displaying PPD file properties

```
// Displays postscript level and path for each PPD file found in a new text frame

var sPPD = "";
var docRef = documents.add();
var x = 30;
var y = (docRef.height - 30);

var iLength = PPDFileList.length;
if (iLength > 20)
    iLength = 20;

for(var i=0; i<iLength; i++) {
    var ppdRef = PPDFileList[i];
    sPPD = ppdRef.name;
    sPPD += "\r\tPS Level ";
    var ppdInfoRef = ppdRef.PPDInfo;
    sPPD += ppdInfoRef.languageLevel;
    sPPD += "\r\tPath: ";
    sPPD += ppdInfoRef.PPDFilePath;

    var textRef = docRef.textFrames.add();
    textRef.textRange.characterAttributes.size = 8;
    textRef.contents = sPPD;
    textRef.top = (y);
    textRef.left = x;
    redraw();

    if( (y-=(textRef.height)) <= 30 ) {
        y = (docRef.height - 30);
        x += 150;
    }
}
```

PPDFileInfo and related screen information

```
// Displays in a new text frame, the postscript level, file paths, screens, and
// screen spot information for first 10 PPD files found

var sPPD = "";
var docRef = documents.add();
var x = 30;
var y = (docRef.height - 30);

var iLength = PPDFileList.length;
if (iLength > 10)
    iLength = 10;
for(var i=0; i<iLength; i++) {
    var ppdRef = PPDFileList[i];
    sPPD = ppdRef.name;
    sPPD += "\r\tPS Level ";
    var ppdInfoRef = ppdRef.PPDInfo;
    sPPD += ppdInfoRef.languageLevel;
    sPPD += "\r\tPath: ";
    sPPD += ppdInfoRef.PPDFilePath;

    sPPD += "\r\tScreens:\r";
    var iScreens = ppdInfoRef.screenList.length;
    for(var c=0; c<iScreens; c++) {
        var screenRef = ppdInfoRef.screenList[c];
        sPPD += "\t\t";
        sPPD += screenRef.name;
        var screenInfoRef = screenRef.screenInfo;
        sPPD += ", Angle = ";
        sPPD += screenInfoRef.angle;
        sPPD += ", Frequency = ";
        sPPD += screenInfoRef.frequency;
        sPPD += "\r";
    }

    sPPD += "\r\tScreenSpots:\r";
    var iScreenSpots = ppdInfoRef.screenSpotFunctionList.length;
    for(var n=0; n<iScreenSpots; n++) {
        var screenSpotRef = ppdInfoRef.screenSpotFunctionList[n];
        sPPD += "\t\t";
        sPPD += screenSpotRef.name;
        sPPD += ", spotFunction: ";
        sPPD += screenSpotRef.spotFunction;
        sPPD += "\r";
    }

    var textRef = docRef.textFrames.add();
    textRef.textRange.characterAttributes.size = 8;
    textRef.contents = sPPD;
    textRef.top = (y);
    textRef.left = x;
    redraw();

    y-=(textRef.height);
}
```

Preferences

Specifies the preferred options for AutoCAD, FreeHand, PDF, and Photoshop files.

Preferences properties

Property	Value type	What it is
`AutoCADFileOptions`	`OpenOptionsAutoCAD`	Read-only. Options to use when opening or placing an AutoCAD file.
`FreeHandFileOptions`	`OpenOptionsFreeHand`	Read-only. Options to use when opening or placing a FreeHand file.
`parent`	object	Read-only. The parent of this object.
`PDFFileOptions`	`PDFFileOptions`	Read-only. Options to use when opening or placing a PDF file.
`PhotoshopFileOptions`	`PhotoshopFileOptions`	Read-only. Options to use when opening or placing a Photoshop file.
`typename`	`string`	Read-only. The class name of the referenced object.

Preferences methods

Method	Parameter type	Returns	What it does
`getBooleanPreference (key)`	`string`	`boolean`	Gets the boolean value of a given application preference.
`getIntegerPreference (key)`	`string`	`integer`	Gets the integer value of a given application preference.
`getRealPreference (key)`	`string`	`double`	Gets the real-number value of a given application preference.
`getStringPreference (key)`	`string`	`string`	Gets the string value of a given application preference.
`removePreference (key)`	`string`	Nothing	Deletes a given application preference.
`setBooleanPreference (key, value)`	`string` `boolean`	Nothing	Sets the boolean value of a given application preference.
`setIntegerPreference (key, value)`	`string` `integer`	Nothing	Sets the integer value of a given application preference.

Method	Parameter type	Returns	What it does
`setRealPreference (key, value)`	`string` `double`	Nothing	Sets the real-number value of a given application preference.
`setStringPreference (key, value)`	`string` `string`	Nothing	Sets the string value of a given application preference.

PrintColorManagementOptions

Information used for color management of the document.

PrintColorManagementOptions properties

Property	Value type	What it is
`colorProfileMode`	`PrintColorProfile`	The color management profile mode. Default: `PrintColorProfile.SOURCEPROFILE`
`intent`	`PrintColorIntent`	The color management intent type. Default: `PrintColorIntent.RELATIVECOLORIMETRIC`
`name`	`string`	The color management profile name.
`typename`	`string`	Read-only. The class name of the object.

Managing colors for printing

```
// Creates a new document, adds symbols, then creates a
// PrintColorManagementOptions object and assigns it
// to a PrintOptions object, then prints with each color intent

// Add some symbol items to a new document
var docRef = documents.add();
var y = docRef.height - 30;
for(var i=0; i<(docRef.symbols.length); i++) {

    symbolRef = docRef.symbols[i];
    symbolItemRef1 = docRef.symbolItems.add(symbolRef);
    symbolItemRef1.top = y;
    symbolItemRef1.left = 100;
    y -= (symbolItemRef1.height + 10);
}
redraw();

var colorOptions = new PrintColorManagementOptions();
var options = new PrintOptions();
options.colorManagementOptions = colorOptions;
colorOptions.name = "ColorMatch RGB";

// Print the current document once for each color intent.
colorOptions.intent = PrintColorIntent.ABSOLUTECOLORIMETRIC;
docRef.print(options);

colorOptions.intent = PrintColorIntent.PERCEPTUALINTENT;
docRef.print(options);

colorOptions.intent = PrintColorIntent.RELATIVECOLORIMETRIC;
docRef.print(options);

colorOptions.intent = PrintColorIntent.SATURATIONINTENT;
docRef.print(options);
```

PrintColorSeparationOptions

Information about the color separations to be used in printing the document.

PrintColorSeparationOptions properties

Property	Value type	What it is
`colorSeparationMode`	`PrintColorSeparationMode`	The color separation type. Default: `PrintColorSeparationMode.COMPOSITE`
`convertSpotColors`	`boolean`	If `true`, all spot colors should be converted to process colors. Default: `false`
`inkList`	`array` of `Ink`	The list of inks for color separation.
`overPrintBlack`	`boolean`	If `true`, overprint in black. Default: `false`
`typename`	`string`	Read-only. The class name of the object.

Managing color separations for printing

```
// Creates a new document with symbol items
// and prints document with each separation option

// Add some symbol items to a new document
var docRef = documents.add();
var y = docRef.height - 30;
for(var i=0; i<(docRef.symbols.length); i++) {
    symbolRef = docRef.symbols[i];
    symbolItemRef1 = docRef.symbolItems.add(symbolRef);
    symbolItemRef1.top = y;
    symbolItemRef1.left = 100;
    y -= (symbolItemRef1.height + 10);
}
// Print with various separation options
var sepOptions = new PrintColorSeparationOptions();
var options = new PrintOptions();
options.colorSeparationOptions = sepOptions;

sepOptions.convertSpotColors = true;
sepOptions.overPrintBlack = true;
sepOptions.colorSeparationMode = PrintColorSeparationMode.COMPOSITE;
docRef.print(options);

sepOptions.colorSeparationMode = PrintColorSeparationMode.INRIPSEPARATION;
docRef.print(options);

sepOptions.convertSpotColors = false;
sepOptions.overPrintBlack = false;
sepOptions.colorSeparationMode = PrintColorSeparationMode.HOSTBASEDSEPARATION;
docRef.print(options);
```

PrintCoordinateOptions

Information about the media and associated printing parameters.

PrintCoordinateOptions properties

Property	Value type	What it is
`emulsion`	`boolean`	If `true`, flip artwork horizontally. Default: `false`
`fitToPage`	`boolean`	If `true`, proportionally scale the artwork to fit on media. Default: `false`
`horizontalScale`	`number (double)`	The horizontal scaling factor expressed as a percentage (100 = 100%). Range: 1.0 to 10000.0. Default: 100.0
`orientation`	`PrintOrientation`	The artwork orientation. Default: `PrintOrientation.PORTRAIT`
`position`	`PrintPosition`	The artwork position on media. Default: `PrintPosition.TRANSLATECENTER`
`tiling`	`PrintTiling`	The page tiling mode. Default: `PrintTiling.TILESINGLEFULLPAGE`
`typename`	`string`	Read-only. The class name of the object.
`verticalScale`	`number (double)`	The vertical scaling factor expressed as a percentage (100 = 100%) Range: 1.0 to 10000.0. Default: 100.0

Managing print coordinates

```
   // Creates a new document with symbol items that extend
// off the page then print with each print orientation

var docRef = documents.add();
var y = 500;
var x = -70
if(docRef.symbols.length > 0){
    for(var i=0; i<5; i++) {
        symbolRef = docRef.symbols[0];
        symbolItemRef1 = docRef.symbolItems.add(symbolRef);
        symbolItemRef1.top = y;
        symbolItemRef1.left = x;
        x += 30;
    }
    redraw();
    // Print it with various Coordinate Options
    var coordinateOptions = new PrintCoordinateOptions();
    var options = new PrintOptions();
    options.coordinateOptions = coordinateOptions;

    coordinateOptions.emulsion = true; // reverse from right to left
    coordinateOptions.fitToPage = true;   // fit artwork to page size
    coordinateOptions.orientation = PrintOrientation.LANDSCAPE;
```

```
    docRef.print(options);
    coordinateOptions.emulsion = false;
    coordinateOptions.fitToPage = false;
    coordinateOptions.orientation = PrintOrientation.PORTRAIT;
    coordinateOptions.horizontalScale = 50;
    coordinateOptions.verticalScale = 50;
    docRef.print(options);
}
```

Printer

Associates an available printer with printer information. To request a list of printers, you must first have a document open or an error is returned.

Printer properties

Property	Value type	What it is
`name`	`string`	The printer name.
`printerInfo`	`PrinterInfo`	The printer information.
`typename`	`string`	Read-only. The class name of the object.

PrinterInfo

Configuration information about a printer.

PrinterInfo properties

Property	Value type	What it is
`binaryPrintingSupport`	`boolean`	If `true`, the printer supports binary printing.
`colorSupport`	`PrinterColorMode`	The printer color capability.
`customPaperSupport`	`boolean`	If `true`, the printer supports custom paper size.
`customPaperTransverseSupport`	`boolean`	If `true`, the printer supports custom paper transverse.
`deviceResolution`	`number (double)`	The printer default resolution.
`inRIPSeparationSupport`	`boolean`	If `true`, the printer supports InRIP color separation.
`maxDeviceResolution`	`number (double)`	The printer maximum device resolution.
`maxPaperHeight`	`number (double)`	Custom paper's maximum height.
`maxPaperHeightOffset`	`number (double)`	Custom paper's maximum height offset.
`maxPaperWidth`	`number (double)`	Custom paper's maximum width.
`maxPaperWidthOffset`	`number (double)`	Custom paper's maximum width offset.
`minPaperHeight`	`number (double)`	Custom paper's minimum height.
`minPaperHeightOffset`	`number (double)`	Custom paper's minimum height offset.
`minPaperWidth`	`number (double)`	Custom paper's minimum width.
`minPaperWidthOffset`	`number (double)`	Custom paper's minimum width offset.
`paperSizes`	`array` of `Paper`	The list of supported paper sizes.
`postScriptLevel`	`PrinterPostScriptLevelEnum`	The PostScript Language level.

Property	Value type	What it is
printerType	PrinterTypeEnum	The printer type.
typename	string	Read-only. The class name of the object.

Finding available printers

```
// Displays a list of available printers in a new text frame

var docRef = documents.add();
var textRef = docRef.textFrames.add();

var iCount = printerList.length;
textRef.contents += "Printers...\r";
for( var i=0; i<iCount; i++ ) {
    textRef.contents += printerList[i].name;
    textRef.contents += "\r\t";
}
textRef.top = 600;
textRef.left = 200;
redraw();
```

PrintFlattenerOptions

Contains flattening options for use when Illustrator outputs artwork that contains transparency into a non-native format.

PrintFlattenerOptions properties

Property	Value type	What it is
`clipComplexRegions`	`boolean`	If `true`, complex regions should be clipped. Default: `false`
`convertStrokesToOutlines`	`boolean`	If `true`, convert all strokes to outlines. Default: `false`
`convertTextToOutlines`	`boolean`	If `true`, all text is converted to vector paths; preserves the visual appearance of type. Default: `false`
`flatteningBalance`	`number (long)`	The flattening balance. Range: 0.0 to 100.0. Default: 100.0
`gradientResolution`	`number (double)`	The gradient resolution in dots per inch (dpi). Range: 1.0 to 9600.0. Default: 300.0
`overprint`	`PDFOverprint`	Whether to preserve, discard, or simulate overprinting. Default: `PDFOverprint.PRESERVEPDFOVERPRINT`
`rasterizationResolution`	`number (double)`	The rasterization resolution in dots per inch (dpi). Range: 1.0 to 9600.0. Default: 300.0
`typename`	`string`	Read-only. The class name of the object.

Setting print flattening

```
// Creates a new document, adds symbols to the document
// then prints with a range of flattener balance settings

var docRef = documents.add();
var y = docRef.height - 30;
for(var i=0; i<(docRef.symbols.length); i++) {

    symbolRef = docRef.symbols[i];
    symbolItemRef1 = docRef.symbolItems.add(symbolRef);
    symbolItemRef1.top = y;
    symbolItemRef1.left = 100;
    y -= (symbolItemRef1.height + 10);
}
redraw();
// Create PrintFlattenerOptions object and assign to a PrintOptions object
var flatOpts = new PrintFlattenerOptions();
var printOpts = new PrintOptions();
printOpts.flattenerOptions = flatOpts;
// Set other print options
printOpts.ClipComplexRegions = true;
printOpts.GradientResoultion = 360;
printOpts.RasterizatonResotion = 360;

// Print the current document with flattening balance increments of 20
var i;
for(i=0; i<=100; i+=20) {
    flatOpts.flatteningBalance = i;
    activeDocument.print(printOpts);
}
```

PrintFontOptions

Contains information about font downloading and substitution for the fonts used for printing the document.

PrintFontOptions properties

Property	Value type	What it is
`downloadFonts`	`PrintFontDownloadMode`	The font download mode. Default: `PrintFontDownloadMode.DOWNLOADSUBSET`
`fontSubstitution`	`FontSubstitutionPolicy`	The font substitution policy. Default: `FontSubstitutionPolicy.SUBSTITUTEOBLIQUE`
`typename`	`string`	Read-only. The class name of the object.

Printing with font options

```
// Creates a new document, adds text then prints with specified font options.

var docRef = documents.add();
var pathRef = docRef.pathItems.rectangle(500,300,400,400);
var textRef = docRef.textFrames.areaText(pathRef);
textRef.contents = "Text example";
//Create PrintFontOptions object and assign to a PrintOptions object
var fontOpts = new PrintFontOptions();
var printOpts = new PrintOptions();
printOpts.fontOptions = fontOpts;
//Set some font options
fontOpts.downloadFonts = PrintFontDownloadMode.DOWNLOADNONE;
fontOpts.fontSubstitution = FontSubstitutionPolicy.SUBSTITUTEDEVICE;

// print it
activeDocument.print(printOpts);
```

PrintJobOptions

Contains information about how the job is to be printed.

PrintJobOptions properties

Property	Value type	What it is
`artboardRange`	`string`	The artboard range to be printed if `printAllArtboards` is false. Default: 1-
`bitmapResolution`	`number (double)`	The bitmap resolution. Minimum: 0.0. Default: 0.0
`collate`	`boolean`	If `true`, collate print pages. Default: `false`
`copies`	`number (long)`	The number of copies to print. Minimum: 1. Default: 1
`designation`	`PrintArtworkDesignation`	The layers/objects to be printed. Default: `PrintArtworkDesignation.VISIBLEPRINTABLELAYERS`
`file`	`File`	The file to which to print.
`name`	`string`	The print job name.
`printAllArtboards`	`boolean`	Indicates whether to print all artboards. Default: `true`
`printArea`	`PrintingBounds`	The printing bounds. Default: `PrintingBounds.ARTBOARDBOUNDS`
`printAsBitmap`	`boolean`	If `true`, print as bitmap. Default: `false`
`reversePages`	`boolean`	If `true`, print pages in reverse order. Default: `false`
`typename`	`string`	Read-only. The class name of the object.

Printing with job options

```
// Creates a new document with layers containing visible, printable,
// non visible and non printable items then prints with each designation
// to view effects of using different job options

var docRef = documents.add();
var textRef_0 = docRef.layers[0].textFrames.add();
textRef_0.contents = "Visible and Printable";
textRef_0.top = 600;
textRef_0.left = 200;

var layerRef_1 = docRef.layers.add();
var textRef_1 = layerRef_1.textFrames.add();
textRef_1.contents = "Visible and Non-Printable";
textRef_1.top = 500;
textRef_1.left = 250;
layerRef_1.printable = false;

var layerRef_2 = docRef.layers.add();
var textRef_2 = layerRef_2.textFrames.add();
textRef_2.contents = "Non-Visible";
textRef_2.top = 400;
textRef_2.left = 300;
layerRef_2.visible = false;
redraw();

// Print with various job options
var printJobOptions= new PrintJobOptions();
var options = new PrintOptions();
options.jobOptions = printJobOptions;

printJobOptions.designation = PrintArtworkDesignation.ALLLAYERS;
printJobOptions.reverse = true;
docRef.print(options);

printJobOptions.collate = false;
printJobOptions.designation = PrintArtworkDesignation.VISIBLELAYERS;
printJobOptions.reverse = false;
docRef.print(options);

printJobOptions.designation = PrintArtworkDesignation.VISIBLEPRINTABLELAYERS;
var docPath = new File("~/printJobTest1.ps");
printJobOptions.file = docPath;
docRef.print(options);
```

PrintOptions

Contains information about all printing options including flattening, color management, coordinates, fonts, and paper.

PrintOptions properties

Property	Value type	What it is
`colorManagementOptions`	`PrintColorManagementOptions`	The printing color management options.
`colorSeparationOptions`	`PrintColorSeparationOptions`	The printing color separation options.
`coordinateOptions`	`PrintCoordinateOptions`	The printing coordinate options.
`flattenerOptions`	`PrintFlattenerOptions`	The printing flattener options.
`flattenerPreset`	`string`	The transparency flattener preset name.
`fontOptions`	`PrintFontOptions`	The printing font options.
`jobOptions`	`PrintJobOptions`	The printing job options.
`pageMarksOptions`	`PrintPageMarksOptions`	The printing page marks options.
`paperOptions`	`PrintPaperOptions`	The paper options.
`postScriptOptions`	`PrintPostScriptOptions`	The printing PostScript options.
`PPDName`	`string`	The PPD name.
`printerName`	`string`	The printer name.
`printPreset`	`string`	The print style.

Setting print options

```
// Creates a new document, adds symbols, specifies a variety of print options,
// assigns each print option to a PrintOptions object,
// then prints with those options

// Create a new document and add some symbol items
var docRef = documents.add();
var y = docRef.height - 30;
for(var i=0; i<(docRef.symbols.length); i++) {
    symbolRef = docRef.symbols[i];
    symbolItemRef1 = docRef.symbolItems.add(symbolRef);
    symbolItemRef1.top = y;
    symbolItemRef1.left = 100;
    y -= (symbolItemRef1.height + 10);
}
redraw();

// Create multiple options and assign to PrintOptions
var options = new PrintOptions();

var colorOptions = new PrintColorManagementOptions();
colorOptions.name = "ColorMatch RGB";
colorOptions.intent = PrintColorIntent.SATURATIONINTENT;
options.colorManagementOptions = colorOptions;

var printJobOptions= new PrintJobOptions();
printJobOptions.designation = PrintArtworkDesignation.ALLLAYERS;
printJobOptions.reverse = true;
options.jobOptions = printJobOptions;

var coordinateOptions = new PrintCoordinateOptions();
coordinateOptions.fitToPage = true;
options.coordinateOptions = coordinateOptions;

var flatOpts = new PrintFlattenerOptions();
flatOpts .ClipComplexRegions = true;
flatOpts .GradientResoultion = 60;
flatOpts .RasterizatonResotion = 60;
options.flattenerOptions = flatOpts;

// Print with options
docRef.print(options);
```

PrintPageMarksOptions

The options for printing page marks.

PrintPageMarksOptions properties

Property	Value type	What it is
`bleedOffsetRect`	`array` of 4 numbers	The bleed offset rectangle.
`colorBars`	`boolean`	If `true`, enable printing of color bars. Default: `false`
`marksOffsetRect`	`array` of 4 numbers	The page marks offset rectangle.
`pageInfoMarks`	`boolean`	If `true`, page info marks printing is enabled. Default: `false`
`pageMarksType`	`PageMarksTypes`	The page marks style. Default: `PageMarksType.Roman`
`registrationMarks`	`boolean`	If `true`, registration marks should be printed. Default: `false`
`trimMarks`	`boolean`	If `true`, trim marks should be printed. Default: `false`
`trimMarksWeight`	`number (double)`	Stroke weight of trim marks. Minimum: 0.0. Default: `0.125`
`typename`	`string`	Read-only. The class name of the object.

Setting page mark printing options

```
// Creates a PrintPageMarksOptions object, assigns it
// to a PrintOptions object, then prints the current document.

var docRef = activeDocument;
var pageMarkOptions= new PrintPageMarksOptions();
var options = new PrintOptions();
options.pageMarksOptions = pageMarkOptions;

pageMarkOptions.colorBars = true;
pageMarkOptions.pageInfoMarks = true;
pageMarkOptions.registrationMarks = true;
pageMarkOptions.trimMarks = true;
docRef.print(options);
```

PrintPaperOptions

Information about the paper to be used in the print job.

PrintPaperOptions properties

Property	Value type	What it is
`height`	`number (double)`	The custom height (in points) for using the custom paper. Default: 0.0
`name`	`string`	The paper's name.
`offset`	`number (double)`	Custom offset (in points) for using the custom paper. Default: 0.0
`transverse`	`boolean`	If `true`, transverse the artwork (rotate 90 degrees) on the custom paper. Default: `false`
`typename`	`string`	Read-only. The class name of the object.
`width`	`number (double)`	The custom width (in points) for using the custom paper. Default: 0.0

Setting print paper options

```
// Creates a new document, adds a path item, applies a graphic style
// then prints with specified paper options

var docRef = documents.add();
var pathRef = docRef.pathItems.rectangle(600, 200, 200, 200);
docRef.graphicStyles[1].applyTo(pathRef);

var paperOpts = new PrintPaperOptions;
var printOpts = new PrintOptions;
printOpts.paperOptions = paperOpts;

var printerCount = printerList.length;
if (printerCount > 0){
    // Print with the 1st paper from the 1st printer
    for (var i = 0; i < printerList.length; i++)
        if (printerList[i].printerInfo.paperSizes.length > 0)
            var printerRef = printerList[i];
    var paperRef = printerRef.printerInfo.paperSizes[0];
    if (printerRef.printerInfo.paperSizes.length > 0){
        paperOpts.name = paperRef.name;
        printOpts.printerName = printerRef.name;

        docRef.print(printOpts);
    }
}
```

PrintPostScriptOptions

Options for printing to a PostScript printer.

PrintPostScriptOptions properties

Property	Value type	What it is
`binaryPrinting`	`boolean`	If `true`, printing should be in binary mode. Default: `false`
`compatibleShading`	`boolean`	If `true`, use PostScript Level 1-compatible gradient and gradient mesh printing. Default: `false`
`forceContinuousTone`	`boolean`	If `true`, force continuous tone. Default: `false`
`imageCompression`	`PostScriptImageCompressionType`	The image compression type. Default: `PostScriptImageCompressionType.IMAGECOMPRESSIONNONE`
`negativePrinting`	`boolean`	If `true`, print in negative mode. Default: `false`
`postScriptLevel`	`PrinterPostScriptLevelEnum`	The PostScript language level. Default: `PrinterPostScriptLevelEnum.LEVEL2`
`shadingResolution`	`number (double)`	The shading resolution. Range: 1.0 to 9600.0 Default: 300.0
`typename`	`string`	Read-only. The class name of the object.

Setting PostScript printing options

```
// Prints current document with various postscript levels

// Create new postscript options object, assign to print options
var psOpts = new PrintPostScriptOptions();
var printOpts = new PrintOptions();
printOpts.postScriptOptions = psOpts;
// Assign PS level, print
psOpts.postScriptLevel = PrinterPostScriptLevelEnum.PSLEVEL2;
activeDocument.print(printOpts);

psOpts.postScriptLevel = PrinterPostScriptLevelEnum.PSLEVEL3;
activeDocument.print(printOpts);
```

RasterEffectOptions

Specifies raster effects settings for the document. All properties are optional.

RasterEffectOptions properties

Property	Value type	What it is
`antiAliasing`	`boolean`	If `true`, the image should be antialiased. Default: `false`
`clippingMask`	`boolean`	If `true`, a clipping mask is created for the image. Default: `false`
`colorModel`	`RasterizationColorModel`	The color model for the rasterization. Default: `RasterizationColorModel.DEFAULTCOLORMODEL`
`convertSpotColors`	`boolean`	If `true`, all spot colors are converted to process colors for the image. Default: `false`
`padding`	`number (double)`	The amount of white space (in points) to be added around the object during rasterization. Default: .0
`resolution`	`number (double)`	The rasterization resolution in dots per inch (dpi). Range: 72.0 to 2400.0. Default: 300.0
`transparency`	`boolean`	If `true`, the image should use transparency. Default: `false`

RasterItem

A bitmap art item in a document. A script can create a raster item from an external file, or by copying an existing raster item with the `duplicate` method.

RasterItem properties

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout.
`bitsPerChannel`	`number (long)`	Read-only. The number of bits per channel.
`blendingMode`	`BlendModes`	The blend mode used when compositing an object.
`boundingBox`	`array` of 4 numbers	The dimensions of the placed art item regardless of transformations.
`channels`	`number (long)`	Read-only. The number of channels.
`colorants`	`array` of `string`	Read-only. The colorants used in the raster art.
`colorizedGrayscale`	`boolean`	Read-only. If `true`, the raster art is a colorized grayscale image.
`contentVariable`	`Variable`	The content variable bound to the item.
`controlBounds`	`array` of 4 numbers	Read-only. The bounds of the object including stroke width and controls.
`editable`	`boolean`	Read-only. If `true`, this item is editable.
`embedded`	`boolean`	If `true`, the raster art item is embedded in the illustration.
`file`	`File`	Read-only. The file containing the artwork.
`geometricBounds`	`array` of 4 numbers	Read-only. The bounds of the object excluding stroke width.
`height`	`number (double)`	The height of the group item.
`hidden`	`boolean`	If `true`, this item is hidden.
`imageColorSpace`	`ImageColorSpace`	Read-only. The color space of the raster image.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layer`	`Layer`	Read-only. The layer to which this item belongs.
`left`	`number (double)`	The position of the left side of the item (in points, measured from the left side of the page).
`locked`	`boolean`	If `true`, this item is locked.
`matrix`	`Matrix`	The transformation matrix of the placed artwork.
`name`	`string`	The name of this item.

Property	Value type	What it is
note	string	The note assigned to this item.
opacity	number (double)	The opacity of the object. Range: 0.0 to 100.0
overprint	boolean	If true, the raster art overprints.
parent	Layer or GroupItem	Read-only. The parent of this object.
position	array of 2 numbers	The position (in points) of the top left corner of the rasterItem object in the format [x, y]. Does not include stroke weight.
selected	boolean	If true, this item is selected.
sliced	boolean	If true, the item sliced. Default: false
status	RasterLinkState	Status of the linked image.
tags	Tags	Read-only. The tags contained in this item.
top	number (double)	The position of the top of the item (in points, measured from the bottom of the page).
transparent	boolean	Read-only. If true, the raster art is transparent.
typename	string	Read-only. The class name of the referenced object.
uRL	string	The value of the Adobe URL tag assigned to this item.
visibilityVariable	Variable	The visibility variable bound to the item.
visibleBounds	array of 4 numbers	Read-only. The visible bounds of the item including stroke width.
width	number (double)	The width of the item.
wrapInside	boolean	If true, the text frame object should be wrapped inside this object.
wrapOffset	number (double)	The offset to use when wrapping text around this object.
wrapped	boolean	If true, wrap text frame objects around this object (text frame must be above the object).
zOrderPosition	number	Read-only. The position of this item within the stacking order of the group or layer (parent) that contains the item.

RasterItem methods

Method	Parameter type	Returns	What it does
`colorize (rasterColor)`	`Color`	Nothing	Colorizes the raster item with a CMYK or RGB Color.
`duplicate ([relativeObject] [,insertionLocation])`	object `ElementPlacement`	`RasterItem`	Creates a duplicate of the selected object.
`move (relativeObject, insertionLocation)`	object `ElementPlacement`	`RasterItem`	Moves the object.
`remove ()`		Nothing	Deletes this object.
`resize (scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])`	`number (double)` `number (double)` `boolean` `boolean` `boolean` `boolean` `number (double)` `Transformation`	Nothing	Scales the art item 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 (double)` `boolean` `boolean` `boolean` `boolean` `Transformation`	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the `angle` value is positive, clockwise if the value is negative.
`trace ()`		`PluginItem`	Converts the raster art for this object to vector art, using default options. Reorders the raster art into the source art of a plug-in group, and converts it into a group of filled and/or stroked paths that resemble the original image. Creates and returns a `pluginItem` object that references a `tracingObject` object.

Method	Parameter type	Returns	What it does

`transform`
`(transformationMatrix`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,transformAbout])`
	
	`Matrix`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Transforms the art item by applying a transformation matrix.

`translate`
`([deltaX]`
`[,deltaY]`
`[,transformObjects]`
`[,transformFillPatterns]`
`[,transformFillGradients]`
`[,transformStrokePatterns])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
		Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.

`zOrder` `(zOrderCmd)`	 `ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

RasterItems

A collection of `RasterItem` objects.

RasterItems properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

RasterItems methods

Method	Parameter type	Returns	What it does
`getByName (name)`	`string`	`RasterItem`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`RasterItem`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Creating a raster item

```
// Creates a new raster item in a new document from a raster file
// jpgFilePath contains the full path and file name of a jpg file

function createRasterItem(jpgFilePath) {
    var rasterFile = File(jpgFilePath);
    var myDoc = app.documents.add();
    var myPlacedItem = myDoc.placedItems.add();
    myPlacedItem.file = rasterFile;
    myPlacedItem.position = Array( 0, myDoc.height );
    myPlacedItem.embed();
}
```

Finding and examining a raster item

```
// Examines the color space of the first raster item in the document and displays
// result in ESTK console

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

    switch ( rasterArt.imageColorSpace ) {
        case ImageColorSpace.CMYK:
            $.writeln("The color space of the first raster item is CMYK");
            break;

        case ImageColorSpace.RGB:
            $.writeln("The color space of the first raster item is RGB");
            break;

        case ImageColorSpace.GRAYSCALE:
            $.writeln("The color space of the first raster item is GRAYSCALE");
            break;
    }
}
```

RasterizeOptions

Specifies options that may be supplied when rasterizing artwork. All properties are optional.

RasterizeOptions properties

Property	Value type	What it is
`antiAliasingMethod`	`AntiAliasingMethod`	The type of antialiasing method. Default: `AntiAliasingMethod.ARTOPTIMIZED`
`backgroundBlack`	`boolean`	If `true`, the rasterization is done against a black background (instead of white). Default: `false`
`clippingMask`	`boolean`	If `true`, a clipping mask should be created for the image. Default: `false`
`colorModel`	`RasterizationColorModel`	The color model for the rasterization. Default: `RasterizationColorModel.DEFAULTCOLORMODEL`
`convertSpotColors`	`boolean`	If `true`, spot colors should be converted to process colors for the image. Default: `false`
`convertTextToOutlines`	`boolean`	If `true`, all text is converted to outlines before rasterization. Default: `false`
`includeLayers`	`boolean`	If `true`, the resulting image incorporates layer attributes (like opacity and blend mode). Default: `false`
`padding`	`number (double)`	The amount of white space (in points) to be added around the object during rasterization. Default: .0
`resolution`	`number (double)`	The rasterization resolution in dots per inch (dpi). Range: 72.0 to 2400.0. Default: 300.0
`transparency`	`boolean`	If `true`, the image should use transparency. Default: `false`

RGBColor

An RGB color specification, used to apply an RGB color to a layer or art item.

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.

RGBColor properties

Property	Value type	What it is
`blue`	`number (double)`	The blue color value. Range: 0.0 to 255.0
`green`	`number (double)`	The green color value. Range: 0.0 to 255.0
`red`	`number (double)`	The red color value. Range: 0.0 to 255.0
`typename`	`string`	Read-only. The class name of the referenced object.

Setting an RGB color

```
// Sets the default fill color in the current document to yellow.

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

    newRGBColor.red = 255;
    newRGBColor.green = 255;
    newRGBColor.blue = 0;
    app.activeDocument.defaultFillColor = newRGBColor;
}
```

Screen

Associates a color separation screen with information to be used for printing.

Screen properties

Property	Value type	What it is
`name`	`string`	The color separation screen name.
`screenInfo`	`ScreenInfo`	The color separation screen information.
`typename`	`string`	Read-only. The class name of the object.

ScreenInfo

Contains information about the angle and frequency of the color separation screen to be used for printing.

ScreenInfo properties

Property	Value type	What it is
`angle`	`number (double)`	The screen's angle in degrees.
`defaultScreen`	`boolean`	If `true`, it is the default screen.
`frequency`	`number (double)`	The screen's frequency.
`typename`	`string`	Read-only. The class name of the object.

Getting screen information

```
// Displays in a new text frame, the name, angle and frequency
// of each screen list item

var sInfo = "";
var docRef = documents.add();
if(PPDFileList.length == 0){
    var sInfo = "\r\t\tEmpty PPDFileList"
}
else{
    var ppdRef = PPDFileList[0];
    var ppdInfoRef = ppdRef.PPDInfo;
    sInfo += "\r\t\tScreen Objects for 1st PPD File:\r";
    sInfo += "\t\t" + ppdRef.name;
    var iScreens = ppdInfoRef.screenList.length;
    if(iScreens > 0){
        for(var c=0; c<iScreens; c++) {
            var screenRef = ppdInfoRef.screenList[c];
            sInfo += "\r\t\t";
            sInfo += screenRef.name;

            var screenInfoRef = screenRef.screenInfo;
            sInfo += ", Angle = ";
            sInfo += screenInfoRef.angle;
            sInfo += ", Frequency = ";
            sInfo += screenInfoRef.frequency;
            sInfo += "\r";
        }
    }
    else{
        sInfo += "\r\t\tEmpty ScreenList";
    }
}
var textRef = docRef.textFrames.add();
textRef.textRange.characterAttributes.size = 12;
textRef.contents = sInfo;
textRef.top = 600;
textRef.left = 30;
redraw();
```

ScreenSpotFunction

Contains information about a color separation screen spot function, including its definition in PostScript language code.

ScreenSpotFunction properties

Property	Value type	What it is
`name`	`string`	The color separation screen spot function name.
`spotFunction`	`string`	The spot function expressed in PostScript commands.
`typename`	`string`	Read-only. The class name of the object.

Finding screen spot functions

```
// Displays in a new text frame, the screen spot functions for the 1st PPD file.

var docRef = documents.add();
if(PPDFileList.length == 0){
    var sInfo = "\r\t\tEmpty PPDFileList"
}
else{
    var ppdRef = PPDFileList[0];
    var ppdInfoRef = ppdRef.PPDInfo;
    var sInfo  = "\r\t\tScreenSpotFunctions for 1st PPD File:\r";
    sInfo += "\t\t" + ppdRef.name + "\r";
    var iScreenSpots = ppdInfoRef.screenSpotFunctionList.length;
    if(iScreenSpots > 0 ){
        for(var n=0; n<iScreenSpots; n++) {
            var screenSpotRef = ppdInfoRef.screenSpotFunctionList[n];
            sInfo += "\t\t";
            sInfo += screenSpotRef.name;
            sInfo += ", spotFunction: ";
            sInfo += screenSpotRef.spotFunction;
            sInfo += "\r";
        }
    }
    else{
        sInfo += "\t\tEmpty ScreenSpotFunctionList";
    }
}
var textRef = docRef.textFrames.add();
textRef.textRange.characterAttributes.size = 12;
textRef.contents = sInfo;
textRef.top = 600;
textRef.left = 30;
redraw();
```

Spot

A custom color definition contained in a SpotColor object.

If no properties are specified when creating a spot, default values are provided. However, if specifying the color, you must use the same color space as the document, either CMYK or RGB. Otherwise, an error results. The new spot is added to the end of the swatches list in the Swatches palette.

Spot properties

Property	Value type	What it is
`color`	`Color`	The color information for this spot color.
`colorType`	`ColorModel`	The color model for this custom color.
`name`	`string`	The spot color's name.
`parent`	`Document`	Read-only. The document that contains this spot color.
`spotKind`	`SpotColorKind`	Read-only. The kind of spot color (RGB, CMYK or LAB). This is the name of the color kind contained in the spot object.
`typename`	`string`	Read-only. The class name of the referenced object.

Spot methods

Method	Parameter type	Returns	What it does
`getInternalColor ()`		Color components	Gets the internal color of a spot.
`remove ()`		Nothing	Deletes this object.

Creating a new spot color

```
// Creates a new spot color in the current document, then applies an 80% tint to the
color

if ( app.documents.length > 0 ){
    var doc = app.activeDocument;
    // Create the new spot
    var newSpot = doc.spots.add();
    // Define the new color value
    var newColor = new CMYKColor();
    newColor.cyan = 35;
    newColor.magenta = 0;
    newColor.yellow = 50;
    newColor.black = 0;
    // Define a new SpotColor with an 80% tint
    // of the new Spot's color. The spot color can then
    // be applied to an art item like any other color.
    newSpot.name = "Pea-Green";
    newSpot.colorType = ColorModel.SPOT;
    newSpot.color = newColor;
    var newSpotColor = new SpotColor();
    newSpotColor.spot = newSpot;
    newSpotColor.tint = 80;
}
```

SpotColor

Color class used to apply the color value of a spot at a specified tint value. Can be used in any property that takes a color object.

SpotColor properties

Property	Value type	What it is
`spot`	`Spot`	A reference to the spot color object that defines the color.
`tint`	`number (double)`	The tint of the color. Range: 0.0 to 100.0
`typename`	`string`	Read-only. The class name of the referenced object.

Spots

A collection of `SpotColor` objects in a document.

Spots properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

Spots methods

Method	Parameter type	Returns	What it does
`add ()`		`Spot`	Creates a new object.
`getByName (name)`	`string`	`Spot`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`Spot`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Removing spot colors

```
// Deletes all spots colors from the current document

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

Creating and applying spot colors

```
// Defines and applies a new spot color in the current document then applies the color
to
// the first path item

if ( app.documents.length > 0 && app.activeDocument.pathItems.length > 0 ) {
    // Define the new color value
    newRGBColor = new RGBColor();
    newRGBColor.red = 255;
    newRGBColor.green = 0;
    newRGBColor.blue = 0;

    // Create the new spot
    var newSpot = app.activeDocument.spots.add();
    // Define the new SpotColor as 80% of the RGB color
    newSpot.name = "Scripted Red spot";
    newSpot.tint = 80;
    newSpot.color = newRGBColor;

    // Apply a 50% tint of the new spot color to the frontmost path item.

    // Create a spotcolor object, set the tint value,
    var newSpotColor = new SpotColor();
    newSpotColor.spot = newSpot;
    newSpotColor.tint = 50;
    // Use the spot color to set the fill color
    var frontPath = app.activeDocument.pathItems[0];
    frontPath.filled = true;
    frontPath.fillColor = newSpotColor;
}
```

Story

A contiguous block of text as specified by a text range. A story can contain one or more text frames; if there is more than one, the multiple text frames are linked together to form a single story.

Story properties

Property	Value type	What it is
`characters`	`Characters`	Read-only. All the characters in this story.
`insertionPoints`	`InsertionPoints`	Read-only. All the insertion points in this story.
`length`	`number (long)`	Read-only. The number of characters in the story.
`lines`	`Lines`	Read-only. All the lines in this story.
`paragraphs`	`Paragraphs`	Read-only. All the paragraphs in this story.
`parent`	object	Read-only. The object's container.
`textFrames`	`TextFrameItems`	Read-only. The text frame items in this story.
`textRange`	`TextRange`	Read-only. The text range of the story.
`textRanges`	`TextRanges`	Read-only. All the text ranges in the story.
`textSelection`	`array` of `TextRange`	Read-only. The selected text ranges in the story.
`typename`	`string`	Read-only. The class name of the object.
`words`	`Words`	Read-only. All the words in the story.

Threading text frames into stories

```
// Creates 1 story that flows through 2 text frames and another story that
// is displayed in a 3rd text frame

// Create a new document and add 2 area TextFrames
var docRef = documents.add();
var itemRef1 = docRef.pathItems.rectangle(600, 200, 50, 30);
var textRef1 = docRef.textFrames.areaText(itemRef1);
textRef1.selected = true;

// create 2nd text frame and link it the first
var itemRef2 = docRef.pathItems.rectangle(550, 300, 50, 200);
var textRef2 = docRef.textFrames.areaText(itemRef2, TextOrientation.HORIZONTAL,
textRef1);
textRef2.selected = true;

// Add enough text to the 1st TextFrame to
// cause it to flow to the 2nd TextFrame.
textRef1.contents = "This is two text frames linked together as one story";
redraw();

// Create a 3rd text frame and count the stories
var textRef3 = docRef.textFrames.add();
textRef3.contents = "Each unlinked textFrame adds a new story."
textRef3.top = 650;
textRef3.left = 200;
redraw();
```

Stories

A collection of `Story` objects in a document.

Stories properties

Property	Value type	What it is
`length`	`number`	Read-only. Number of elements in the collection.
`parent`	object	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the object.

Stories methods

Method	Parameter type	Returns	What it does
`index (itemKey)`	`string, number`	`Story`	Gets an element from the collection.

Swatch

A color swatch definition contained in a document. The swatches correspond to the swatch palette in the Illustrator user interface. A script can create a new swatch. The swatch can hold all types of color data, such as pattern, gradient, CMYK, RGB, gray, and spot.

Swatch properties

Property	Value type	What it is
`color`	`Color`	The color information for this swatch.
`name`	`string`	The swatch's name.
`parent`	`Document`	Read-only. The document that contains this swatch.
`typename`	`string`	Read-only. The class name of the referenced object.

Swatch methods

Method	Parameter type	Returns	What it does
`remove ()`		Nothing	Deletes this object.

Modifying a swatch

```
// Changes the name of the last swatch

if ( app.documents.length > 0 && app.activeDocument.swatches.length > 0 ) {
    var lastIndex = app.activeDocument.swatches.length - 1;
    var lastSwatch = app.activeDocument.swatches[lastIndex];
    lastSwatch.name = "TheLastSwatch";
}
```

Swatches

A collection of `Swatch` objects in a document.

Swatches properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

Swatches methods

Method	Parameter type	Returns	What it does
`add ()`		`Swatch`	Creates a new `Swatch` object.
`getByName (name)`	`string`	`Swatch`	Gets the first element in the collection with the specified name.
`getSelected ()`		List of `Swatch`	Gets selected swatches in the document.
`index (itemKey)`	`string, number`	`Swatch`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Finding and deleting a swatch

```
// Deletes swatch 4 from the current document

if ( app.documents.length > 0 ) {
   if (app.activeDocument.swatches.length > 4)
   {
       swatchToDelete = app.activeDocument.swatches[3];
       swatchToDelete.remove();
   }
}
```

SwatchGroup

A group of `Swatch` objects.

SwatchGroup properties

Property	Value type	What it is
`name`	`string`	The name of the swatch group.
`parent`	object	Read-only. The object that contains the symbol object.
`typename`	`string`	Read-only. The class name of the referenced object.

SwatchGroup methods

Method	Parameter type	Returns	What it does
`addSpot (spot)`	`Spot`	Nothing	Adds a spot swatch to the swatch group.
`addSwatch (swatch)`	`Swatch`	Nothing	Adds a swatch to the swatch group.
`getAllSwatches ()`		List of `Swatch`	Gets a list of all swatches in the swatch group.
`remove ()`		Nothing	Deletes this object.
`removeAll ()`		Nothing	Deletes all elements in the collection.

SwatchGroups

A collection of `SwatchGroup` objects.

SwatchGroups properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

SwatchGroups methods

Method	Parameter type	Returns	What it does
`add`		`SwatchGroup`	Creates a swatch group.
`getByName (name)`	`string`	`SwatchGroup`	Gets the first element in the collection with the specified name.
`removeAll ()`		Nothing	Deletes all elements in the collection.

Symbol

An art item 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. Instances of `Symbol` in a document are associated with `SymbolItem` objects, which store the art object properties.

Symbol properties

Property	Value type	What it is
`name`	`string`	The symbol's name.
`parent`	object	Read-only. The object that contains the symbol object.
`typename`	`string`	Read-only. The class name of the referenced object.

Symbol methods

Method	Parameter type	Returns	What it does
`duplicate ()`		`Symbol`	Create a duplicate of this object.
`remove ()`		Nothing	Deletes this object.

SymbolItem

An art item made reusable by adding it to the Symbols palette. A `SymbolItem` is linked to the `Symbol` from which it was created and changes if you modify the associated `Symbol` object.

SymbolItem properties

Property	Value type	What it is
`artworkKnockout`	`KnockoutState`	Is this object used to create a knockout, and if so, what kind of knockout.
`blendingMode`	`BlendModes`	The blend mode used when compositing an object.
`controlBounds`	`array` of 4 numbers	Read-only. The bounds of the object including stroke width and controls.
`editable`	`boolean`	Read-only. If `true`, this item is editable.
`geometricBounds`	`array` of 4 numbers	Read-only. The bounds of the object excluding stroke width.
`height`	`number (double)`	The height of the group item.
`hidden`	`boolean`	If `true`, this item is hidden.
`isIsolated`	`boolean`	If `true`, this object is isolated.
`layer`	`Layer`	Read-only. The layer to which this item belongs.
`left`	`number (double)`	The position of the left side of the item (in points, measured from the left side of the page).
`locked`	`boolean`	If `true`, this item is locked.
`name`	`string`	The name of this item.
`note`	`string`	The note assigned to this item.
`opacity`	`number (double)`	The opacity of the object. Range: 0.0 to 100.0
`parent`	`Layer` or `GroupItem`	Read-only. The parent of this object.
`position`	`array` of 2 numbers	The position (in points) of the top left corner of the `symbolItem` object in the format [x, y]. Does not include stroke weight.
`selected`	`boolean`	If `true`, this item is selected.
`sliced`	`boolean`	If `true`, the item sliced. Default: `false`
`symbol`	`Symbol`	The symbol that was used to create this `symbolItem`.
`tags`	`Tags`	Read-only. The tags contained in this item.
`top`	`number (double)`	The position of the top of the item (in points, measured from the bottom of the page).

Property	Value type	What it is
typename	string	Read-only. The class name of the referenced object.
uRL	string	The value of the Adobe URL tag assigned to this item.
visibilityVariable	Variable	The visibility variable bound to the item.
visibleBounds	array of 4 numbers	Read-only. The visible bounds of the item including stroke width.
width	number (double)	The width of the item.
wrapInside	boolean	If true, the text frame object should be wrapped inside this object.
wrapOffset	number (double)	The offset to use when wrapping text around this object.
wrapped	boolean	If true, wrap text frame objects around this object (text frame must be above the object).
zOrderPosition	number	Read-only. The position of this item within the stacking order of the group or layer (parent) that contains the item.

SymbolItem methods

Method	Parameter type	Returns	What it does
duplicate ([relativeObject] [,insertionLocation])	 object ElementPlacement	SymbolItem	Creates a duplicate of the selected object.
move (relativeObject, insertionLocation)	 object ElementPlacement	SymbolItem	Moves the object.
remove ()		Nothing	Deletes this object.
resize (scaleX, scaleY [,changePositions] [,changeFillPatterns] [,changeFillGradients] [,changeStrokePattern] [,changeLineWidths] [,scaleAbout])	 number (double) number (double) boolean boolean boolean boolean number (double) Transformation	Nothing	Scales the art item 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 (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`Transformation`
		Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the `angle` value is positive, clockwise if the value is negative.

`transform`
`(transformationMatrix`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,transformAbout])`
	
	`Matrix`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Transforms the art item by applying a transformation matrix.

`translate`
`([deltaX]`
`[,deltaY]`
`[,transformObjects]`
`[,transformFillPatterns]`
`[,transformFillGradients]`
`[,transformStrokePatterns])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
		Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.

`zOrder` `(zOrderCmd)`	 `ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

SymbolItems

A collection of `SymbolItem` objects in the document.

SymbolItems properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

SymbolItems methods

Method	Parameter type	Returns	What it does
`add (symbol)`	`Symbol`	`SymbolItem`	Creates an instance of the specified symbol.
`getByName (name)`	`string`	`SymbolItem`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`SymbolItem`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in the collection.

Creating symbol items

```
// Creates a new document then adds each of
// the documents symbols to the document

var docRef = documents.add();
var y = 750;
var x = 25;
var iCount = docRef.symbols.length;
for(var i=0; i<iCount; i++) {
    symbolRef = docRef.symbols[i];
    symbolItemRef1 = docRef.symbolItems.add(symbolRef);
    symbolItemRef1.top = y;
    symbolItemRef1.left = x;
    y-=(symbolItemRef1.height + 20);
    if( (y) <= 60 ) {
        y = 750;
        x+= 190;
    }
}
```

Symbols

The collection of `Symbol` objects in the document.

Symbols properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

Symbols methods

Method	Parameter type	Returns	What it does
`add (sourceArt, [registrationPoint])`	`PageItem` `SymbolRegistrationPoint`	`Symbol`	Returns a symbol object created from the source art item, any of the following: `CompoundPathItems` `GroupItems` `MeshItems` `NonNativeItems` `PageItems` `PathItems` `RasterItems` `SymbolItems` `TextFrameItems` The default registration point is `SymbolCenterPoint`.
`index (itemKey)`	`string, number`	`Symbol`	Gets an element from the collection.
`getByName (name)`	`string`	`Symbol`	Gets the first element in the collection with the specified name.
`removeAll ()`		Nothing	Deletes all elements in the collection.

Creating a symbol

```
// Creates a path item from each graphic style
// then adds each item as a new symbol

var docRef = documents.add();
var y = 750;
var x =25;

var iCount = docRef.graphicStyles.length;
for(var i=0; i<iCount; i++) {
    var pathRef = docRef.pathItems.rectangle( y, x, 20, 20 );
docRef.graphicStyles[i].applyTo(pathRef);
    // are we at bottom?
    if( (y-=60) <= 60 ) {
        y = 750;  // go back to the top.
        x+= 200
    }
    redraw();
    docRef.symbols.add(pathRef);
}
```

TabStopInfo

Information about the alignment, position, and other details for a tab stop in a `ParagraphAttributes` object.

TabStopInfo properties

Property	Value type	What it is
`alignment`	`TabStopAlignment`	The alignment of the tab stop. Default: `Left`
`decimalCharacter`	`string`	The character used for decimal tab stops. Default: `.`
`leader`	`string`	The leader dot character.
`position`	`number (double)`	The position of the tab stop expressed in points. Default: 0.0
`typename`	`string`	Read-only. The class name of the object.

Displaying tab stop information

```
// Displays tab stop information found in each text frame
// of current document, if any.

docRef = app.activeDocument;
var tabRef;
var sData = "Tab Stops Found \rTabStop Leader\t\tTabStop Position\r";
var textRef = docRef.textFrames;

for( var i=0 ; i < textRef.length; i++ ) {
    // Get all paragraphs in the textFrames
    paraRef = textRef[i].paragraphs;
    for ( p=0 ; p < paraRef.length ; p++ ) {
        // Get para attributes for all textRanges in paragraph
        attrRef = paraRef[p].paragraphAttributes;
        tabRef = attrRef.tabStops;
        if ( tabRef.length > 0 ) {
            for(var t=0; t<tabRef.length; t++){
                sData += "\t" + tabRef[t].leader + "\t\t";
                sData += "\t\t" + tabRef[t].position + "\r";
                } // end for
            } // end if
        } // end for
    } // end for
var newTF = docRef.textFrames.add();
newTF.contents = sData;
newTF.top = 400;
newTF.left = 100;
redraw();
```

Tag

A label associated with a specific piece of artwork. Tags allows you to assign an unlimited number of key-value pairs to any page item in a document.

Tag properties

Property	Value type	What it is
`name`	`string`	The tag's name.
`parent`	object	Read-only. The object that contains this tag.
`typename`	`string`	Read-only. The class name of the referenced object.
`value`	`string`	The data stored in this tag.

Tag methods

Method	Parameter type	Returns	What it does
`remove ()`		Nothing	Deletes this object.

Using tags

```
// Finds the tags associated with the selected art item,
// show names and values in a separate document

if ( app.documents.length > 0 ) {
    doc = app.activeDocument;
    if ( doc.selection.length > 0 ) {
        for ( i = 0; i < selection.length; i++ ) {
            selectedArt = selection[0];
            tagList = selectedArt.tags;
            if (tagList.length == 0) {
                var tempTag = tagList.add();
                tempTag.name = "OneWord";
                tempTag.value = "anything you want";
            }
            // Create a document and add a line of text per tag
            reportDocument = app.documents.add();
            top_offset = 400;
            for ( i = 0; i < tagList.length; i++ ) {
                tagText = tagList[i].value;
                newItem = reportDocument.textFrames.add();
                newItem.contents = "Tag: (" + tagList[i].name +
                    " , " + tagText + ")";
                newItem.position = Array(100, top_offset);
                newItem.textRange.size = 24;
                top_offset = top_offset - 20;
            }
        }
```

```
        }
    }
```

Tags

A collection of `Tag` objects.

Tags properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

Tags methods

Method	Parameter type	Returns	What it does
`add ()`		`Tag`	Creates a new `Tag` object.
`getByName (name)`	`string`	`Tag`	Gets the first element in the collection with the specified name.
`index (itemKey)`	`string, number`	`Tag`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Setting tag values

```
// Adds tags to all RasterItems and PlacedItems in the current document

if ( app.documents.length > 0 ) {
    doc = app.activeDocument;
    if ( doc.placedItems.length + doc.rasterItems.length > 0 ) {
        for ( i = 0; i < doc.pageItems.length; i++ ) {
            imageArt = doc.pageItems[i];
            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/";
            }
        }
    }
    else {
        alert( "No placed or raster items in the document" );
    }
}
```

TextFont

Information about a font in the document, found in a `CharacterAttributes` object.

TextFont properties

Property	Value type	What it is
`family`	`string`	Read-only. The font's family name.
`name`	`string`	Read-only. The font's full name.
`parent`	object	Read-only. The object's container.
`style`	`string`	Read-only. The font's style name.
`typename`	`string`	Read-only. The class name of the object.

Setting the font of text

```
// Sets the font of all the text in the document to the first font

if ( app.documents.length > 0 ) {
    // Iterate through all text art and apply font 0
    for ( i = 0; i< app.activeDocument.textFrames.length; i++) {
        textArtRange = app.activeDocument.textFrames[i].textRange;
        textArtRange.characterAttributes.textFont = app.textFonts[0];
    }
}
```

TextFonts

A collection of `TextFont` objects.

TextFonts properties

Property	Value type	What it is
`length`	`number`	Read-only. Number of elements in the collection.
`parent`	object	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the object.

TextFonts methods

Method	Parameter type	Returns	What it does
`index (itemKey)`	`string, number`	`TextFont`	Get an element from the collection.
`getByName (name)`	`string`	`TextFont`	Get the first element in the collection with the provided name.

Finding fonts

```
// Creates a new A3 sized document and display a list of available fonts until the
document is full.

var edgeSpacing = 10;
var columnSpacing = 230;
var docPreset = new DocumentPreset;
docPreset.width = 1191.0;
docPreset.height = 842.0

var docRef = documents.addDocument(DocumentColorSpace.CMYK, docPreset);
var sFontNames = "";
var x = edgeSpacing;
var y = (docRef.height - edgeSpacing);

var iCount = textFonts.length;
for(var i=0; i<iCount; i++) {
    sFontName = textFonts[i].name;
    sFontName += " ";
    sFontNames = sFontName + textFonts[i].style;

    var textRef = docRef.textFrames.add();
    textRef.textRange.characterAttributes.size = 10;
    textRef.contents = sFontNames;
    textRef.top = y;
    textRef.left = x;

    // check wether the text frame will go off the edge of the document
```

```
    if ((x + textRef.width)> docRef.width){
        textRef.remove();
        iCount = i;
        break;
    }
    else{
        // display text frame
        textRef.textRange.characterAttributes.textFont =
textFonts.getByName(textFonts[i].name);
        redraw();

        if( (y-=(textRef.height)) <= 20 ) {
            y = (docRef.height - edgeSpacing);
            x += columnSpacing;
        }
    }
}
```

TextFrameItem

The basic art item for displaying text. From the user interface, this is text created with the Text tool. There are three types of text art in Illustrator: point text, path text, and area text. The type is indicated by the text frame's kind property.

When you create a text frame, you also create a Story object. However, threading text frames combines the frames into a single story object. To thread frames, use the nextFrame or previousFrame property.

TextFrameItem properties

Property	Value type	What it is
`anchor`	`array` of 2 numbers	The position of the anchor point, the start of the base line for point text.
`antialias`	`TextAntialias`	The type of anti-aliasing to use in the text.
`characters`	`Characters`	Read-only. All the characters in this text frame.
`columnCount`	`number (long)`	The column count in the text frame (area text only).
`columnGutter`	`number (double)`	The column gutter in the text frame (area text only).
`contents`	`string`	The text string.
`contentVariable`	`Variable`	The content variable bound to this text frame item.
`endTValue`	`number (double)`	The end position of text along a path, as a value relative to the path's segments (path text only).
`flowLinksHorizontally`	`boolean`	If `true`, flow text between linked frames horizontally first (area text only).
`insertionPoints`	`InsertionPoints`	Read-only. All the insertion points in this text range.
`kind`	`TextType`	Read-only. The type of a text frame item (area, path or point).
`lines`	`Lines`	Read-only. All the lines in this text frame.
`matrix`	`Matrix`	Read-only. The transformation matrix for this text frame.
`nextFrame`	`TextFrameItem`	The linked text frame following this one.
`opticalAlignment`	`boolean`	If `true`, the optical alignment feature is active.
`orientation`	`TextOrientation`	The orientation of the text.
`paragraphs`	`Paragraphs`	Read-only. All the paragraphs in this text frame.
`parent`	`Layer` or `GroupItem`	Read-only. The parent of this object.
`previousFrame`	`TextFrameItem`	The linked text frame preceding this one.
`rowCount`	`number (long)`	The row count in the text frame (area text only).

Property	Value type	What it is
rowGutter	number (double)	The row gutter in the text frame (area text only).
spacing	number (double)	The amount of spacing.
startTValue	number (double)	The start position of text along a path, as a value relative to the path's segments (path text only).
story	Story	Read-only. The story to which the text frame belongs.
textPath	TextPath	The path item associated with the text frame. Note: Valid only when kind is area or path.
textRange	TextRange	Read-only. The text range of the text frame.
textRanges	TextRanges	Read-only. All the text in this text frame.
textSelection	array of TextRange	Read-only. The selected text range(s) in the text frame.
typename	string	Read-only. The class name of the referenced object.
words	Words	Read-only. All the words in this text frame.

TextFrameItem methods

Method	Parameter type	Returns	What it does
convertAreaObjectToPointObject ()		TextFrameItem	Converts the area-type text frame to a point-type text frame.
convertPointObjectToAreaObject ()		TextFrameItem	Converts the point-type text frame to an area-type text frame.
createOutline ()		GroupItem	Converts the text in the text frame to outlines.
duplicate ([relativeObject] [,insertionLocation])	object ElementPlacement	TextRange	Creates a duplicate of the selected object.
move (relativeObject, insertionLocation)	object ElementPlacement	TextRange	Moves the object.
remove ()		Nothing	Deletes this object.

Method	Parameter type	Returns	What it does

`resize`
`(scaleX,`
`scaleY`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,scaleAbout])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Scales the art item 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 (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`Transformation`
		Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the `angle` value is positive, clockwise if the value is negative.

`transform`
`(transformationMatrix`
`[,changePositions]`
`[,changeFillPatterns]`
`[,changeFillGradients]`
`[,changeStrokePattern]`
`[,changeLineWidths]`
`[,transformAbout])`
	
	`Matrix`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
	`number (double)`
	`Transformation`
		Nothing	Transforms the art item by applying a transformation matrix.

`translate`
`([deltaX]`
`[,deltaY]`
`[,transformObjects]`
`[,transformFillPatterns]`
`[,transformFillGradients]`
`[,transformStrokePatterns])`
	
	`number (double)`
	`number (double)`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
		Nothing	Repositions the art item relative to the current position, where `deltaX` is the horizontal offset and `deltaY` is the vertical offset.

`zOrder` `(zOrderCmd)`	 `ZOrderMethod`	Nothing	Arranges the art item's position in the stacking order of the group or layer (parent) of this object.

Rotate a text art item

```
// Duplicates and rotates the selected text art item 5 times

if ( app.documents.length > 0 ) {
    selectedItems = app.activeDocument.selection;
    // make sure something is selected.
    if ( selectedItems.length > 0 ) {
        // The selection must be a text art item
        if ( selectedItems[0].typename == "TextFrame" ) {
            // Get the parent of the text art so new text art items
            // can be inserted in the same group or layer
            dupSrc = selectedItems[0];
            textContainer = dupSrc.parent;
            // Create 5 new versions of the text art each rotated a bit
            for ( i = 1; i <= 5; i++ ) {
                dupText = dupSrc.duplicate( textContainer,
                       ElementPlacement.PLACEATEND );
                dupText.rotate(180 * i/6);
            }
        }
    }
}
```

TextFrameItems

A collection of `TextFrameItem` objects.

TextFrameItems properties

Property	Value type	What it is
`length`	`number`	Read-only. Number of elements in the collection.
`parent`	object	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the object.

TextFrameItems methods

Method	Parameter type	Returns	What it does
`add ()`		`TextFrameItem`	Creates a point text frame item.
`areaText (textPath [,orientation] [,baseFrame] [,postFix])`	`PathItem` `TextOrientation` `TextFrameItem` `boolean`	`TextFrameItem`	Creates an area text frame item.
`getByName (name)`	`string`	`TextFrameItem`	Gets the first element in the collection with the provided name.
`index (itemKey)`	`string, number`	`TextFrameItem`	Gets an element from the collection.
`pathText (textPath [,startTValue] [,endTValue [,orientation] [,baseFrame] [,postFix])`	`PathItem` `number (double)` `number (double)` `TextOrientation` `TextFrameItem` `boolean`	`TextFrameItem`	Creates an on-path text frame item.
`pointText (anchor [,orientation])`	`array` of 2 numbers `TextOrientation`	`TextFrameItem`	Creates a point text frame item.
`removeAll ()`		Nothing	Deletes all elements in the object.

Creating and modifying text frames

```
// Creates a document with text frames displaying path, area and point
// text, changes the content of each frame then deletes the 2nd frame

// create a new document
var docRef = documents.add();
// create 3 new textFrames (area, line, point)
// Area Text
var rectRef = docRef.pathItems.rectangle(700, 50, 100, 100);
var areaTextRef = docRef.textFrames.areaText(rectRef);
areaTextRef.contents = "TextFrame #1";
areaTextRef.selected = true;

// Line Text
var lineRef = docRef.pathItems.add();
lineRef.setEntirePath( Array(Array(200, 700), Array(300, 550)  ) );
var pathTextRef = docRef.textFrames.pathText(lineRef);
pathTextRef.contents = "TextFrame #2";
pathTextRef.selected = true;

// Point Text
var pointTextRef = docRef.textFrames.add();
pointTextRef.contents = "TextFrame #3";
pointTextRef.top = 700;
pointTextRef.left = 400;
pointTextRef.selected = true;
redraw();

// count the TextFrames
var iCount = docRef.textFrames.length;
var sText = "There are " + iCount + " TextFrames.\r"
sText += "Changing contents of each TextFrame.";

// change the content of each
docRef.textFrames[0].contents = "Area TextFrame.";
docRef.textFrames[1].contents = "Path TextFrame.";
docRef.textFrames[2].contents = "Point TextFrame.";
redraw();

docRef.textFrames[1].remove();
redraw();

// count again
var iCount = docRef.textFrames.length;
```

TextPath

A path or list of paths for area or path text. A path consists of path points that define its geometry.

TextPath properties

Property	Value type	What it is
`area`	`number (double)`	Read-only. The area of this path in square points. If the area is negative, the path is wound counterclockwise. Self-intersecting paths can contain sub-areas that cancel each other out, which makes this value zero even though the path has apparent area.
`blendingMode`	`BlendModes`	The blend mode used when compositing an object.
`clipping`	`boolean`	If `true`, this path should be used as a clipping path.
`closed`	`boolean`	If `true`, this path is closed.
`editable`	`boolean`	Read-only. If `true`, this item is editable.
`evenodd`	`boolean`	If `true`, the even-odd rule should be used to determine insideness.
`fillColor`	`Color`	The fill color of the path.
`filled`	`boolean`	If `true`, the path be filled.
`fillOverprint`	`boolean`	If `true`, the art beneath a filled object should be overprinted.
`guides`	`boolean`	If `true`, this path is a guide object.
`height`	`number (double)`	The height of the group item.
`left`	`number (double)`	The position of the left side of the item (in points, measured from the left side of the page).
`note`	`string`	The note text assigned to the path.
`opacity`	`number (double)`	The opacity of the object. Range: 0.0 to 100.0
`parent`	`Layer` or `GroupItem`	Read-only. The parent of this object.
`pathPoints`	`PathPoints`	Read-only. The path points contained in this path item.
`polarity`	`PolarityValues`	The polarity of the path.
`position`	`array` of 2 numbers	The position (in points) of the top left corner of the `textPathItem` object in the format [x, y]. Does not include stroke weight.
`resolution`	`number (double)`	The resolution of the path in dots per inch (dpi).
`selectedPathPoints`	`PathPoints`	Read-only. All of the selected path points in the path.

Property	Value type	What it is
`strokeCap`	`StrokeCap`	The type of line capping.
`strokeColor`	`Color`	The stroke color for the path.
`stroked`	`boolean`	If `true`, the path should be stroked.
`strokeDashes`	object	Dash lengths. Set to an empty object, `{}`, for a solid line.
`strokeDashOffset`	`number (double)`	The default distance into the dash pattern at which the pattern should be started.
`strokeJoin`	`StrokeJoin`	Type of joints for the path.
`strokeMiterLimit`	`number (double)`	When a default stroke join is set to `mitered`, this property specifies when the join will be converted to beveled (squared-off) by default. The default miter limit of 4 means that when the length of the point reaches four times the stroke weight, the join switches from a miter join to a bevel join. A value of 1 specifies a bevel join. Range: 1 to 500. Default: 4
`strokeOverprint`	`boolean`	If `true`, the art beneath a stroked object should be overprinted.
`strokeWidth`	`number (double)`	Width of the stroke.
`top`	`number (double)`	The position of the top of the item (in points, measured from the bottom of the page).
`typename`	`string`	Read-only. The class name of the referenced object.
`width`	`number (double)`	The width of the item.

TextPath methods

Method	Parameter type	Returns	What it does
`setEntirePath (pathPoints)`	`array` of `[x, y]` coordinate pairs	Nothing	Sets the path using the array of points specified as [x, y] coordinate pairs.

TextRange

A range of text in a specific text art item. `TextRange` gives you access to the text contained in text art items.

TextRange properties

Property	Value type	What it is
`characterAttributes`	`CharacterAttributes`	Read-only. The character properties for the text range.
`characterOffset`	`number (long)`	Offset of the first character.
`characters`	`Characters`	Read-only. All the characters in this text range.
`characterStyles`	`CharacterStyles`	Read-only. All referenced character styles in the text range.
`contents`	`string`	The text string.
`insertionPoints`	`InsertionPoints`	Read-only. All the insertion points in this text range.
`kerning`	`number (long)`	Controls the spacing between two characters, in thousandths of an em. An integer.
`length`	`number (long)`	The length (in characters). Minimum: 0
`lines`	`Lines`	Read-only. All the lines in this text range.
`paragraphAttributes`	`ParagraphAttributes`	Read-only. The paragraph properties for the text range.
`paragraphs`	`Paragraphs`	Read-only. All the paragraphs in this text range.
`paragraphStyles`	`ParagraphStyles`	Read-only. All referenced paragraph styles in the text range.
`parent`	`TextRange`	Read-only. The object's container.
`story`	`Story`	Read-only. The story to which the text range belongs.
`textRanges`	`TextRanges`	Read-only. All of the text in this text range.
`textSelection`	`array` of `TextRange`	Read-only. The selected text ranges in the text range.
`typename`	`string`	Read-only. The class name o f the object.
`words`	`Words`	Read-only. All the words contained in this text range.

TextRange methods

Method	Parameter Type	Returns	What it does
`changeCaseTo (type)`	`CaseChangeType`	Nothing	Changes the capitalization of text.
`deSelect ()`		Nothing	Deselects the text range.
`duplicate ([relativeObject] [,insertionLocation])`	object `ElementPlacement`	`TextRange`	Creates a duplicate of this object.
`move (relativeObject, insertionLocation)`	object `ElementPlacement`	`TextRange`	Moves the object.
`remove ()`		Nothing	Deletes the object.
`select ([addToDocument])`	`boolean`	Nothing	Selects the text range. If `addToDocument` is `true`, adds this to the current selection; otherwise replaces the current selection.

Manipulating text

```
// Changes size of the first character of each word in the
// current document by changing the size attribute of each character

if ( app.documents.length > 0 ) {
    for ( i = 0; i < app.activeDocument.textFrames.length; i++ ) {
        text = app.activeDocument.textFrames[i].textRange;
        for ( j = 0 ; j < text.words.length; j++ ) {
            //each word is a textRange object
            textWord = text.words[j];
            // Characters are textRanges too.
            // Get the first character of each word and increase it's size.
            firstChars = textWord.characters[0];
            firstChars.size = firstChars.size * 1.5;
        }
    }
}
```

TextRanges

A collection of `TextRange` objects.

TextRanges properties

Property	Value type	What it is
`length`	`number`	Read-only. Number of elements in the collection.
`parent`	object	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the object.

TextRanges methods

Method	Parameter type	Returns	What it does
`index (itemKey)`	`string, number`	`TextRange`	Get an element from the collection
`removeAll ()`		Nothing	Deletes all elements in the object.

TracingObject

A tracing object, which associates source raster art item with a vector-art plug-in group created by tracing. Scripts can initiate tracing using `PlacedItem.trace` or `RasterItem.trace`. The resulting `PluginItem` object represents the vector art group, and has this object in its `tracing` property.

A script can force the tracing operation by calling the application's `redraw` method. The operation is asynchronous, so a script should call `redraw` after creating the tracing object, but before accessing its properties or expanding the tracing to convert it to an art item group.

The read-only properties that describe the tracing result have valid values only after the first tracing operation completes. A value of 0 indicates that the operation has not yet been completed.

TracingObject properties

Property	Value type	What it is
`anchorCount`	`number (long)`	Read-only. The number of anchors in the tracing result.
`areaCount`	`number (long)`	Read-only. The number of areas in the tracing result.
`imageResolution`	`number (real)`	Read-only. The resolution of the source image in pixels per inch.
`parent`	object	Read-only. The object's container.
`pathCount`	`number (long)`	Read-only. The number of paths in the tracing result.
`sourceArt`	`PlacedItem` or `RasterItem`	The raster art used to create the associated vector art plug-in group.
`tracingOptions`	`TracingOptions`	The options used to convert the raster artwork to vector art.
`typename`	`string`	Read-only. The class name of the object.
`usedColorCount`	`number (long)`	Read-only. The number of colors used in the tracing result.

TracingObject methods

Method	Parameter type	Returns	What it does
`expandTracing ([viewed])`	`boolean`	`GroupItem`	Converts the vector art into a new group item. The new `GroupItem` object replaces the `PluginItem` object in the document. By default, `viewed` is `false`, and the new group contains only the tracing result (the filled or stroked paths). If `viewed` is `true`, the new group retains additional information that was specified for the viewing mode, such as outlines and overlays. Deletes this object and its associated `PluginItem` object. Any group-level attributes that were applied to the plug-in item are applied to the top level of the new group item.
`releaseTracing ()`		`PlacedItem` or `RasterItem`	Reverts the artwork in the document to the original source raster art and removes the traced vector art. Returns the original object used to create the tracing, and deletes this object and its associated `PluginItem` object.

TracingOptions

A set of options used in converting raster art to vector art by tracing.

TracingOptions properties

Property	Value type	What it is
`cornerAngle`	`number (double)`	The sharpness, in degrees of a turn in the original image that is considered a corner in the tracing result path. Range: 0 to 180
`fills`	`boolean`	If `true`, trace with fills. At least one of `fills` or `strokes` must be `true`.
`ignoreWhite`	`boolean`	If `true`, ignores white fill color.
`livePaintOutput`	`boolean`	If `true`, result is LivePaint art. If `false`, it is classic art. **NOTE:** A script should only set this value in preparation for a subsequent expand operation. Leaving a tracing on the artboard when this property is `true` can lead to unexpected application behavior.
`maxColors`	`number (long)`	The maximum number of colors allowed for automatic palette generation. Used only if `tracingMode` is color or grayscale. Range: 2 to 256
`maxStrokeWeight`	`number (double)`	The maximum stroke weight, when `strokes` is `true`. Range: 0.01 to 100.0
`minArea`	`number (long)`	The smallest feature, in square pixels, that is traced. For example, if it is 4, a feature of 2 pixels wide by 2 pixels high is traced.
`minStrokeLength`	`number (double)`	The minimum length in pixels of features in the original image that can be stroked, when `strokes` is `true`. Smaller features are omitted. Range: 0.0 to 200.0. Default: 20.0
`outputToSwatches`	`boolean`	If `true`, named colors (swatches) are generated for each new color created by the tracing result. Used only if `tracingMode` is color or grayscale.
`palette`	`string`	The name of a color palette to use for tracing. If the empty string, use the automatic palette. Used only if `tracingMode` is color or grayscale.
`parent`	object	Read-only. The object's container.
`pathFitting`	`number (double)`	The distance between the traced shape and the original pixel shape. Lower values create a tighter path fitting. Higher values create a looser path fitting. Range: 0.0 to 10.0

Property	Value type	What it is
`preprocessBlur`	`number (double)`	The amount of blur used during preprocessing, in pixels. Blurring helps reduce small artifacts and smooth jagged edges in the tracing result. Range: 0.0 to 2.0
`preset`	`string`	Read-only. The name of a preset file containing these options.
`resample`	`boolean`	If `true`, resample when tracing. (This setting is not captured in a preset file.) Always `true` when the raster source art is placed or linked.
`resampleResolution`	`number (double)`	The resolution to use when resampling in pixels per inch (ppi). Lower resolution increases the speed of the tracing operation. (This setting is not captured in a preset file.)
`strokes`	`boolean`	If `true`, trace with strokes. At least one of `fills` or `strokes` must be `true`. Used only if `tracingMode` is black-and-white.
`threshold`	`number (long)`	The threshold value of black-and-white tracing. All pixels with a grayscale value greater than this are converted to black. Used only if `tracingMode` is black-and-white. Range: 0 to 255
`tracingMode`	`TracingModeType`	The color mode for tracing.
`typename`	`string`	Read-only. The class name of the object.
`viewRaster`	`ViewRasterType`	The view for previews of the raster image. (This setting is not captured in a preset file.)
`viewVector`	`ViewVectorType`	The view for previews of the vector result. (This setting is not captured in a preset file.)

TracingOptions methods

Method	Parameter type	Returns	What it does
`loadFromPreset (presetName)`	`string`	`boolean`	Loads a set of options from the specified preset, as found in the `Application.tracingPresetList` array.
`storeToPreset (presetName)`	`string`	`boolean`	Saves this set of options in the specified preset. Use a name found in the `Application.tracingPresetList` array, or a new name to create a new preset. For an existing preset, overwrites an unlocked preset and returns `true`. Returns `false` if the preset is locked.

Variable

A document-level variable that can be imported or exported.

A variable is a dynamic object used to create data-driven graphics. For an example, see Dataset. Variables are accessed in Illustrator through the Variables palette.

Variable properties

Property	Value type	What it is
`kind`	`VariableKind`	The variable's type.
`name`	`string`	The name of the variable.
`pageItems`	`PageItems`	Read-only. All of the artwork in the variable.
`parent`	object	Read-only. The object that contains the variable.
`typename`	`string`	Read-only. The class name of the referenced object.

Variable methods

Method	Parameter type	Returns	What it does
`remove ()`		Nothing	Removes the variable from the collection of variables.

Variables

The collection of `Variable` objects in the document. For an example of how to create variables, see Using variables and datasets.

Variables properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of variables in the document
`parent`	object	Read-only. The object that contains the collection of variables.
`typename`	`string`	Read-only. The class name of the referenced object.

Variables methods

Method	Parameter type	Returns	What it does
`add ()`		`Variable`	Adds a new variable to the collection.
`getByName (name)`	`string`	`Variable`	Get the first element in the collection with the provided name.
`index (itemKey)`	`string, number`	`Variable`	Get an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in the collection.

View

A document view in an Illustrator document, which represents a window view onto a document. Scripts cannot create new views, but can modify some properties of existing views, including the center point, screen mode, and zoom.

View properties

Property	Value type	What it is
`bounds`	`array` of 4 numbers	Read-only. 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`	`Document`	Read-only. The document that contains this view.
`screenMode`	`ScreenMode`	The mode of display for this view.
`typename`	`string`	Read-only. The class name of the referenced object.
`zoom`	`number (double)`	The zoom factor of this view, where 100.0 is 100%.

Setting a view to full screen

```
// Sets the screen mode of the current document to full screen

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

Views

A collection of `View` objects in a document.

Views properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

Views methods

Method	Parameter type	Returns	What it does
`index (itemKey)`	`string, number`	`View`	Gets an element from the collection.

Words

A collection of words in a text item, where each word is a `TextRange` object. The elements are not named; you must access them by index.

Words properties

Property	Value type	What it is
`length`	`number`	Read-only. The number of objects in the collection.
`parent`	object	Read-only. The parent of this object.
`typename`	`string`	Read-only. The class name of the referenced object.

Words methods

Method	Parameter type	Returns	What it does
`add (contents [, relativeObject] [, insertionLocation])`	`string` `TextFrameItem` `ElementPlacement`	`TextRange`	Adds a word to the current document at the specified location. If no location is specified, adds it to the containing text frame after the current word selection or insertion point.
`addBefore (contents)`	`string`	`TextRange`	Adds a word before the current word selection or insertion point.
`index (itemKey)`	`number`	`TextRange`	Gets an element from the collection.
`removeAll ()`		Nothing	Deletes all elements in this collection.

Counting words

```
// Counts all words in current document and stores total in numWords

if ( app.documents.length > 0 ) {
    numWords = 0;
    for ( i = 0; i < app.activeDocument.textFrames.length; i++) {
        numWords += app.activeDocument.textFrames[i].words.length;
    }
}
```

Applying attributes to words

```
// Creates a new magenta color and applies the color to all words meeting a specific
criteria

if ( app.documents.length > 0 && app.activeDocument.textFrames.length > 0 ) {
    // Create the color to apply to the words
    wordColor = new RGBColor();
    wordColor.red = 255;
    wordColor.green = 0;
    wordColor.blue = 255;
    // Set the value of the word to look for
    searchWord1 = "the";
    searchWord2 = "The";
    searchWord3 = "THE";
    // Iterate through all words in the document
    // and color the words that match searchWord
    for ( i = 0; i < app.activeDocument.textFrames.length; i++ ) {
        textArt = activeDocument.textFrames[i];
        for ( j = 0; j < textArt.words.length; j++) {
            word = textArt.words[j];
            if ( word.contents == searchWord1 ||  word.contents == searchWord2 ||
                    word.contents == searchWord3 ) {
                word.filled = true;
                word.fillColor = wordColor;
            }
        }
    }
}
```

2 Scripting Constants

This chapter lists and describes the enumerations defined for use with Illustrator JavaScript properties and methods.

Constant Type	Values		What it means
`AlternateGlyphsForm`			
	`DEFAULTFORM`	`THIRDWIDTH`	
	`TRADITIONAL`	`QUARTERWIDTH`	
	`EXPERT`	`FULLWIDTH`	
	`JIS78FORM`	`PROPORTIONALWIDTH`	
	`JIS83FORM`	`JIS90FORM`	
	`HALFWIDTH`	`JIS04FORM`	
`AntiAliasingMethod`			
	`None`	`TYPEOPTIMIZED`	The type of antialiasing method used in the rasterization. ▶ `None` — No antialiasing is allowed. ▶ `ARTOPTIMIZED` — Optimize for the art object. ▶ `TYPEOPTIMIZED` — Optimize for the type object.
	`ARTOPTIMIZED`		
`ArtClippingOption`			
	`OUTPUTARTBOUNDS`		How the art should be clipped during output. ▶ `OUTPUTARTBOUNDS` — Output size is the size of the artwork. ▶ `OUTPUTARTBOARDBOUNDS` — Output size is the size of the artboard. ▶ `OUTPUTCROPRECTBOUNDS` — Output size is the size of the crop area.
	`OUTPUTARTBOARDBOUNDS`		
	`OUTPUTCROPRECTBOUNDS`		
`AutoCADColors`			
	`Max8Colors`	`Max256Colors`	
	`Max16Colors`	`TrueColors`	

Constant Type	Values		What it means
AutoCADCompatibility			
	AutoCADRelease13	AutoCADRelease18	
	AutoCADRelease14	AutoCADRelease21	
	AutoCADRelease15	AutoCADRelease24	
AutoCADExportFileFormat			
	DXF	DWG	
AutoCADExportOption			
	PreserveAppearance		
	MaximizeEditability		
AutoCADGlobalScaleOption			
	OriginalSize	ScaleByValue	
	FitArtboard		
AutoCADRasterFormat			
	PNG	JPEG	
AutoCADUnit			
	Points	Millimeters	
	Picas	Centimeters	
	Inches	Pixels	
AutoKernType			
	NOAUTOKERN	OPTICAL	
	AUTO	METRICSROMANONLY	
AutoLeadingType			
	BOTTOMTOBOTTOM	TOPTOTOP	
BaselineDirectionType			
	Standard	VerticalRotated	
	TateChuYoko		
BlendAnimationType			
	INBUILD	NOBLENDANIMATION	
	INSEQUENCE		
BlendModes			
	COLORBLEND	LIGHTEN	The blend mode used when compositing an object.
	COLORBURN	LUMINOSITY	
	COLORDODGE	MULTIPLY	
	DARKEN	NORMAL	
	DIFFERENCE	OVERLAY	
	EXCLUSION	SATURATIONBLEND	
	HARDLIGHT	SCREEN	
	HUE	SOFTLIGHT	

Constant Type	Values		What it means
BlendsExpandPolicy			Policy used by FXG file format to expand blends.
	AUTOMATICALLYCONVERTBLENDS		
	RASTERIZEBLENDS		
BurasagariTypeEnum			
	Forced	Standard	
	None		
CaseChangeType			
	LOWERCASE	TITLECASE	
	SENTENCECASE	UPPERCASE	
ColorConversion			
	COLORCONVERSIONREPURPOSE		
	COLORCONVERSIONTODEST		
	None		
ColorConvertPurpose			
	defaultpurpose	exportpurpose	The purpose of color conversion using the `ConvertSampleColor` method of the `Application` class.
	previewpurpose	dummypurpose	
ColorDestination			
	COLORDESTINATIONDOCCMYK		
	COLORDESTINATIONDOCRGB		
	COLORDESTINATIONPROFILE		
	COLORDESTINATIONWORKINGCMYK		
	COLORDESTINATIONWORKINGRGB		
	None		
ColorDitherMethod			
	DIFFUSION	NOREDUCTION	The method used to dither colors in exported GIF and PNG8 images.
	NOISE	PATTERNDITHER	
ColorModel			
	PROCESS	SPOT	
	REGISTRATION		
ColorProfile			
	INCLUDEALLPROFILE	LEAVEPROFILEUNCHANGED	
	INCLUDEDESTPROFILE	None	
	INCLUDERGBPROFILE		
ColorReductionMethod			
	ADAPTIVE	SELECTIVE	The method used to reduce the number of colors in exported GIF and PNG8 images.
	PERCEPTUAL	WEB	

Constant Type	Values		What it means
ColorType			
	CMYK	PATTERN	The color specification for an individual color.
	GRADIENT	RGB	
	GRAY	SPOT	
	NONE		
Compatibility			
	ILLUSTRATOR8	ILLUSTRATOR12	The version of the Illustrator file to create when saving an EPS or Illustrator file
	ILLUSTRATOR9	ILLUSTRATOR13	
	ILLUSTRATOR10	ILLUSTRATOR14	
	ILLUSTRATOR11	ILLUSTRATOR15	
	ILLUSTRATOR16	ILLUSTRATOR17	
	JAPANESEVERSION3	ILLUSTRATOR19	
CompressionQuality			
	AUTOMATICJPEG2000HIGH	JPEG2000LOW	The quality of bitmap compression used when saving a PDF file
	AUTOMATICJPEG2000LOSSLESS	JPEG2000MAXIMUM	
	AUTOMATICJPEG2000LOW	JPEG2000MEDIUM	
	AUTOMATICJPEG2000MAXIMUM	JPEG2000MINIMUM	
	AUTOMATICJPEG2000MEDIUM	JPEGHIGH	
	AUTOMATICJPEG2000MINIMUM	JPEGLOW	
	AUTOMATICJPEGHIGH	JPEGMAXIMUM	
	AUTOMATICJPEGLOW	JPEGMEDIUM	
	AUTOMATICJPEGMAXIMUM	JPEGMINIMUM	
	AUTOMATICJPEGMEDIUM	ZIP4BIT	
	AUTOMATICJPEGMINIMUM	ZIP8BIT	
	JPEG2000HIGH	None	
	JPEG2000LOSSLESS		
CoordinateSystem			
	DOCUMENTCOORDINATESYSTEM		The coordinate system used by Illustrator
	ARTBOARDCOORDINATESYSTEM		
CropOptions			
	Japanese		The style of a document's cropping box
	Standard		
DocumentArtboardLayout			
	GridByRow	RLGridByRow	The layout of in the new document.
	GridByCol	RLGridByCol	
	Row	RLRow	
	Column		
DocumentColorSpace			
	CMYK	RGB	The color space of a document
DocumentLayoutStyle			
	CASCADE	FLOATALL	The layout style for a document.
	HORIZONTALTILE	CONSOLIDATEALL	
	VERTICALTILE		

Constant Type	Values		What it means
DocumentPresetType			
	BasicCMYK BasicRGB Print	Mobile Video Web	The preset types available for new documents.
DocumentPreviewMode			
	DefaultPreview PixelPreview	OverprintPreview	The document preview mode
DocumentRasterResolution			
	ScreenResolution MediumResolution	HighResolution	The preset document raster resolution
DocumentTransparencyGrid			
	TransparencyGridNone TransparencyGridLight TransparencyGridMedium TransparencyGridDark TransparencyGridRed TransparencyGridOrange TransparencyGridGreen TransparencyGridBlue TransparencyGridPurple		Document transparency grid colors
DocumentType			
	EPS ILLUSTRATOR	PDF FXG	The file format used to save a file
DownsampleMethod			
	AVERAGEDOWNSAMPLE BICUBICDOWNSAMPLE	NODOWNSAMPLE SUBSAMPLE	
ElementPlacement			
	INSIDE PLACEATBEGINNING PLACEATEND	PLACEBEFORE PLACEAFTER	
EPSPostScriptLevelEnum			
	LEVEL2 LEVEL3		
EPSPreview			
	BWTIFF COLORTIFF TRANSPARENTCOLORTIFF None		The preview image format used when saving an EPS file

Constant Type	Values		What it means
ExportType			
	FLASH	PNG24	The file format used to export a file
	GIF	PNG8	
	JPEG	SVG	
	Photoshop	TIFF	
	AutoCAD		
FigureStyleType			
	DEFAULTFIGURESTYLE	TABULAR	
	PROPORTIONAL	TABULAROLDSTYLE	
	PROPORTIONALOLDSTYLE		
FiltersPreservePolicy			
	EXPANDFILTERS		The filters preserve policy used by the FXG file format.
	KEEPFILTERSEDITABLE		
	RASTERIZEFILTERS		
FlashExportStyle			
	ASFLASHFILE	LAYERSASSYMBOLS	The method used to convert Illustrator images when exporting files
	LAYERSASFRAMES	TOFILES	
	LAYERSASFILES		
FlashExportVersion			
	FlashVersion1	FlashVersion6	Version for exported SWF file
	FlashVersion2	FlashVersion7	
	FlashVersion3	FlashVersion8	
	FlashVersion4	FlashVersion9	
	FlashVersion5		
FlashImageFormat			
	LOSSLESS		The format used to store flash images
	LOSSY		
FlashJPEGMethod			
	Optimized		The method used to store JPEG images
	Standard		
FlashPlaybackSecurity			
	PlaybackLocal		
	PlaybackNetwork		
FontBaselineOption			
	NORMALBASELINE		
	SUPERSCRIPT		
	SUBSCRIPT		
FontCapsOption			
	ALLCAPS	NORMALCAPS	
	ALLSMALLCAPS	SMALLCAPS	

Constant Type	Values		What it means
FontOpenTypePositionOption			
	DENOMINATOR	OPENTYPESUBSCRIPT	
	NUMERATOR	OPENTYPESUPERSCRIPT	
	OPENTYPEDEFAULT		
FontSubstitutionPolicy			
	SUBSTITUTEDEVICE		
	SUBSTITUTEOBLIQUE		
	SUBSTITUTETINT		
FXGVersion			
	VERSION1PT0		The FXG file-format version.
	VERSION2PT0		
GradientsPreservePolicy			
	AUTOMATICALLYCONVERTGRADIENTS		The gradients preserve policy used by the FXG file format.
	KEEPGRADIENTSEDITABLE		
GradientType			
	LINEAR		The type of gradient
	RADIAL		
ImageColorSpace			
	CMYK	Separation	The color space of a raster item or an exported file
	Grayscale	DeviceN	
	RGB	Indexed	
	LAB		
InkPrintStatus			
	CONVERTINK		
	ENABLEINK		
	DISABLEINK		
InkType			
	BLACKINK	MAGENTAINK	
	CUSTOMINK	YELLOWINK	
	CYANINK		
JavaScriptExecutionMode			
	BeforeRunning		
	OnRuntimeError		
	never		
Justification			
	CENTER	FULLJUSTIFYLASTLINECENTER	The alignment or justification for a paragraph of text
	LEFT	FULLJUSTIFYLASTLINELEFT	
	RIGHT	FULLJUSTIFYLASTLINERIGHT	
	FULLJUSTIFY		

Constant Type	Values		What it means
KinsokuOrderEnum			
	PUSHIN		
	PUSHOUTONLY		
	PUSHOUTFIRST		
KnockoutState			
	DISABLED	INHERITED	The type of knockout to use on a page item
	ENABLED	Unknown	
LanguageType			
	BOKMALNORWEGIAN	JAPANESE	
	BRAZILLIANPORTUGUESE	NYNORSKNORWEGIAN	
	BULGARIAN	OLDGERMAN	
	CANADIANFRENCH	POLISH	
	CATALAN	RUMANIAN	
	CHINESE	RUSSIAN	
	CZECH	SERBIAN	
	DANISH	SPANISH	
	DUTCH	STANDARDFRENCH	
	DUTCH2005REFORM	STANDARDGERMAN	
	ENGLISH	STANDARDPORTUGUESE	
	FINNISH	SWEDISH	
	GERMAN2006REFORM	SWISSGERMAN	
	GREEK	SWISSGERMAN2006REFORM	
	HUNGARIAN	TURKISH	
	ICELANDIC	UKENGLISH	
	ITALIAN	UKRANIAN	
LayerOrderType			
	TOPDOWN		
	BOTTOMUP		
LibraryType			
	IllustratorArtwork	GraphicStyles	Illustrator library type
	Swatches	Symbols	
	Brushes		
MonochromeCompression			
	CCIT3	None	The type of compression to use on a monochrome bitmap item when saving a PDF file
	CCIT4	RUNLENGTH	
	MONOZIP		
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		

Constant Type	Values		What it means
PageMarksTypes			
	Japanese		
	Roman		
PathPointSelection			
	ANCHORPOINT	NOSELECTION	Which points, if any, of a path are selected
	LEFTDIRECTION	RIGHTDIRECTION	
	LEFTRIGHTPOINT		
PDFBoxType			
	PDFARTBOX	PDFCROPBOX	
	PDFBLEEDBOX	PDFMEDIABOX	
	PDFBOUNDINGBOX	PDFTRIMBOX	
PDFChangesAllowedEnum			
	CHANGE128ANYCHANGES	CHANGE40ANYCHANGES	
	CHANGE128COMMENTING	CHANGE40COMMENTING	
	CHANGE128EDITPAGE	CHANGE40PAGELAYOUT	
	CHANGE128FILLFORM	CHANGE40NONE	
	CHANGE128NONE		
PDFCompatibility			
	ACROBAT4	ACROBAT7	The version of the Acrobat file format to create when saving a PDF file
	ACROBAT5	ACROBAT8	
	ACROBAT6		
PDFOverprint			
	DISCARDPDFOVERPRINT		
	PRESERVEPDFOVERPRINT		
PDFPrintAllowedEnum			
	PRINT128HIGHRESOLUTION		
	PRINT128LOWRESOLUTION		
	PRINT128NONE		
	PRINT40HIGHRESOLUTION		
	PRINT40NONE		
PDFTrimMarkWeight			
	TRIMMARKWEIGHT0125		
	TRIMMARKWEIGHT05		
	TRIMMARKWEIGHT025		
PDFXStandard			
	PDFXNONE	PDFX32002	
	PDFX1A2001	PDFX32003	
	PDFX1A2003	PDFX42007	

Constant Type	Values	What it means
PerspectiveGridType	OnePointPerspectiveGridType TwoPointPerspectiveGridType ThreePointPerspectiveGridType InvalidPerspectiveGridType	
PerspectiveGridPlaneType	GRIDLEFTPLANETYPE GRIDRIGHTPLANETYPE GRIDFLOORPLANETYPE INVALIDGRIDPLANETYPE	
PhotoshopCompatibility	Photoshop6 Photoshop8	
PointType	CORNER SMOOTH	The type of path point selected
PolarityValues	NEGATIVE POSITIVE	
PostScriptImageCompressionType	IMAGECOMPRESSIONNONE RLE JPEG	
PrintArtworkDesignation	ALLLAYERS VISIBLELAYERS VISIBLEPRINTABLELAYERS	
PrintColorIntent	ABSOLUTECOLORIMETRIC PERCEPTUALINTENT RELATIVECOLORIMETRIC SATURATIONINTENT	
PrintColorProfile	CUSTOMPROFILE OLDSTYLEPROFILE PRINTERPROFILE SOURCEPROFILE	
PrintColorSeparationMode	COMPOSITE HOSTBASEDSEPARATION INRIPSEPARATION	

Constant Type	Values		What it means
PrinterColorMode			
	BLACKANDWHITEPRINTER		
	GRAYSCALEPRINTER		
	COLORPRINTER		
PrinterPostScriptLevelEnum			
	PSLEVEL1		
	PSLEVEL2		
	PSLEVEL3		
PrinterTypeEnum			
	NONPOSTSCRIPTPRINTER		
	POSTSCRIPTPRINTER		
	Unknown		
PrintFontDownloadMode			
	DOWNLOADNONE		
	DOWNLOADCOMPLETE		
	DOWNLOADSUBSET		
PrintingBounds			
	ARTBOARDBOUNDS		
	ARTWORKBOUNDS		
PrintOrientation			
	AUTOROTATE		The artwork printing orientation.
	LANDSCAPE	REVERSELANDSCAPE	
	PORTRAIT	REVERSEPORTRAIT	
PrintPosition			
	TRANSLATEBOTTOM	TRANSLATERIGHT	
	TRANSLATEBOTTOMLEFT	TRANSLATETOP	
	TRANSLATEBOTTOMRIGHT	TRANSLATETOPLEFT	
	TRANSLATECENTER	TRANSLATETOPRIGHT	
	TRANSLATELEFT		
PrintTiling			
	TILEFULLPAGES		
	TILESINGLEFULLPAGE		
	TILEIMAGEABLEAREAS		
RasterizationColorModel			
	DEFAULTCOLORMODEL		The color model for the rasterization.
	BITMAP		
	GRAYSCALE		
RasterLinkState			
	DATAFROMFILE		The status of a raster item's linked image if the image is stored externally
	DATAMODIFIED		
	NODATA		

Constant Type	Values		What it means
RulerUnits			
	Centimeters	Qs	The default measurement units for the rulers of a document
	Inches	Pixels	
	Millimeters	Unknown	
	Picas		
	Points		
SaveOptions			
	DONOTSAVECHANGES		Save options provided when closing a document
	SAVECHANGES		
	PROMPTTOSAVECHANGES		
ScreenMode			
	DESKTOP		The mode of display for a view
	MULTIWINDOW		
	FULLSCREEN		
SpotColorKind			
	SpotCMYK		The custom color kind of a spot color
	SpotLAB		
	SpotRGB		
StrokeCap			
	BUTTENDCAP		The type of line capping for a path stroke
	ROUNDENDCAP		
	PROJECTINGENDCAP		
StrokeJoin			
	BEVELENDJOIN		The type of joints for a path stroke
	ROUNDENDJOIN		
	MITERENDJOIN		
StyleRunAlignmentType			
	bottom	icfTop	
	center	ROMANBASELINE	
	icfBottom	top	
SVGCSSPropertyLocation			
	ENTITIES	STYLEATTRIBUTES	How should the CSS properties of the document be included in an exported SVG file
	PRESENTATIONATTRIBUTES	STYLEELEMENTS	
SVGDocumentEncoding			
	ASCII		How should the text in the document be encoded when exporting an SVG file
	UTF8		
	UTF16		

Constant Type	Values		What it means
SVGDTDVersion			
	SVG1_0	SVGTINY1_1	SVB version compatibility for exported files
	SVG1_1	SVGTINY1_1PLUS	
	SVGBASIC1_1	SVGTINY1_2	
SVGFontSubsetting			
	ALLGLYPHS	GLYPHSUSEDPLUSENGLISH	What font glyphs should be included in exported SVG files
	COMMONENGLISH	GLYPHSUSEDPLUSROMAN	
	COMMONROMAN	None	
	GLYPHSUSED		
SVGFontType			
	CEFFONT		Types for fonts included in exported SVG files
	SVGFONT		
	OUTLINEFONT		
SymbolRegistrationPoint			
	SYMBOLTOPLEFTPOINT		Registration points for symbols
	SYMBOLTOPMIDDLEPOINT		
	SYMBOLTOPRIGHTPOINT		
	SYMBOLMIDDLELEFTPOINT		
	SYMBOLCENTERPOINT		
	SYMBOLMIDDLERIGHTPOINT		
	SYMBOLBOTTOMLEFTPOINT		
	SYMBOLBOTTOMMIDDLEPOINT		
	SYMBOLBOTTOMRIGHTPOINT		
TabStopAlignment			
	Center	Left	The alignment of a tab stop
	Decimal	Right	
TextAntialias			
	CRISP		The type of text anti-aliasing in a text art item
	NONE		
	SHARP		
	STRONG		
TextOrientation			
	HORIZONTAL		The orientation of text in a text art item
	VERTICAL		
TextPreservePolicy			
	AUTOMATICALLYCONVERTTEXT		The text-preserve policy used by the FXG file format.
	OUTLINETEXT		
	KEEPTEXTEDITABLE		
	RASTERIZETEXT		
TextType			
	AREATEXT		The type of text art displayed by this object
	POINTTEXT		
	PATHTEXT		

Constant Type	Values	What it means
TIFFByteOrder		
	IBMPC MACINTOSH	The byte order to use for an exported TIFF file.
TracingModeType		
	TRACINGMODEBLACKANDWHITE TRACINGMODECOLOR TRACINGMODEGRAY	
Transformation		
	BOTTOM BOTTOMLEFT BOTTOMRIGHT CENTER DOCUMENTORIGIN LEFT RIGHT TOP TOPLEFT TOPRIGHT	The point to use as the anchor point about which an object is rotated, resized, or transformed
TrappingType		
	IGNOREOPAQUE NORMALTRAPPING OPAQUE TRANSPARENT	
UserInteractionLevel		
	DISPLAYALERTS DONTDISPLAYALERTS	User interface settings
VariableKind		
	GRAPH IMAGE TEXTUAL Unknown VISIBILITY	What type of variables are included in the document
ViewRasterType		
	TRACINGVIEWRASTERADJUSTEDIMAGE TRACINGVIEWRASTERNOIMAGE TRACINGVIEWRASTERORIGINALIMAGE TRACINGVIEWRASTERTRANSPARENTIMAGE	The raster visualization mode for tracing.
ViewVectorType		
	TRACINGVIEWVECTORNOTRACINGRESULT TRACINGVIEWVECTOROUTLINES TRACINGVIEWVECTOROUTLINESWITHTRACING TRACINGVIEWVECTORTRACINGRESULT	The vector visualization mode for tracing.
WariChuJustificationType		
	Center Left Right WARICHUAUTOJUSTIFY WARICHUFULLJUSTIFY WARICHUFULLJUSTIFYLASTLINECENTER WARICHUFULLJUSTIFYLASTLINELEFT WARICHUFULLJUSTIFYLASTLINERIGHT	

Constant Type	Values		What it means
`ZOrderMethod`			
	`BRINGFORWARD`	`SENDBACKWARD`	The method used to arrange an art item's position in the stacking order of its parent group or layer, as specified with the `zOrder` method
	`BRINGTOFRONT`	`SENDTOBACK`	