ADOBE® PHOTOSHOP® CS3

JAVASCRIPT SCRIPTING REFERENCE

Adobe®

© 2007 Adobe Systems Incorporated. All rights reserved.

Adobe® Creative Suite® 3 Photoshop® JavaScript Scripting Reference for Windows® and Macintosh®.

NOTICE: All information contained herein is the property of Adobe Systems Incorporated. No part of this publication (whether in hardcopy or electronic form) may be reproduced or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of Adobe Systems Incorporated. The software described in this document is furnished under license and may only be used or copied in accordance with the terms of such license.

This publication and the information herein is furnished AS IS, is subject to change without notice, and should not be construed as a commitment by Adobe Systems Incorporated. Adobe Systems Incorporated assumes no responsibility or liability for any errors or inaccuracies, makes no warranty of any kind (express, implied, or statutory) with respect to this publication, and expressly disclaims any and all warranties of merchantability, fitness for particular purposes, and noninfringement of third party rights.

Any references to company names in sample templates are for demonstration purposes only and are not intended to refer to any actual organization.

Adobe®, the Adobe logo, Acrobat®, GoLive®,InDesign®, Illustrator®, Photoshop® are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Apple®, Mac OS®, and Macintosh® are trademarks of Apple Computer, Inc., registered in the United States and other countries. Microsoft®, and Windows®. are either registered trademarks or trademarks of Microsoft Corporation in the United States and other countries. JavaScript™ and all Java-related marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. UNIX® is a registered trademark of The Open Group.

All other trademarks are the property of their respective owners.

If this guide is distributed with software that includes an end user agreement, this guide, as well as the software described in it, is furnished under license and may be used or copied only in accordance with the terms of such license. Except as permitted by any such license, no part of this guide may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of Adobe Systems Incorporated. Please note that the content in this guide is protected under copyright law even if it is not distributed with software that includes an end user license agreement.

The content of this guide is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Adobe Systems Incorporated. Adobe Systems Incorporated assumes no responsibility or liability for any errors or inaccuracies that may appear in the informational content contained in this guide.

Adobe Systems Incorporated, 345 Park Avenue, San Jose, California 95110, USA.

Contents

1 Introduction

This reference describes the objects and methods in the Adobe® Photoshop® CS® 3 JavaScript™ type library. A companion document, *Photoshop CS3 Scripting Guide*, describes basic scripting concepts and the Photoshop object model. This document provides reference details of the Photoshop object model, and additional information on JavaScript-specific features.

Adobe Photoshop CS3 uses ExtendScript, Adobe's extended implementation of JavaScript. See JavaScript support in Adobe Photoshop CS3 for additional information.

This book contains the following sections:

- This introduction, which describes scripting support in Adobe Photoshop CS3, and lists changes to the JavaScript interface since the previous release.
- JavaScript Object Reference, which provides a complete reference for all Photoshop DOM objects and commands.
- Scripting Constants, which lists all enumerations used in the Photoshop type library.

JavaScript support in Adobe Photoshop CS3

For a JavaScript file to be recognized by Photoshop as a valid script file, it must use either a `.js` or a `.jsx` extension.

On the Mac OS, there is no difference in the way scripts with the two extensions function. On Windows, if the script files is opened from inside Photoshop, there is no difference between using the `.js` and `.jsx` extension. However, if the script is launched by double-clicking on it, a script with the `.js` extension is interpreted with the Microsoft JScript engine, and it cannot launch Adobe Photoshop CS3. For Windows, using the `.jsx` extension is preferrable, since it interprets the script with the ExtendScript engine.

All of the Adobe Creative Suite 3 applications, including Adobe Photoshop CS3, use ExtendScript, Adobe's extended implementation of JavaScript. ExtendScript files are distinguished by the `.jsx` extension. ExtendScript offers all standard JavaScript features, plus additional features and utilities, such as:

- A debugging environment (the ExtendScript Toolkit)
- A localization utility
- Tools that allow you to combine scripts and direct them to particular applications
- Platform-independent file and folder representation

Many of the JavaScript objects and methods use objects defined in ExtendScript, such as the `File` object, the `Folder` object, and the `UnitValue` object. For that reason, using the `.jsx` extension for your script files is preferable. For details of these and additional features, see the *JavaScript Tools Guide CS3*.

Executing scripts

The Adobe Photoshop CS3 interface includes a Scripts menu (**File > Scripts**) which provides quick and easy access to your JavaScripts. Scripts can be listed directly as menu items that run when you select them, or you can navigate to and run any JavaScript in your file system.

If Adobe Photoshop CS3 encounters an error during script execution, it displays the error message.

Installing scripts

To install a JavaScript in the Scripts menu, place it in the Scripts folder (**Photoshop CS3/Presets/Scripts**). The names of the scripts in the Scripts folder, without the file name extension, will be displayed in the Scripts menu. Any number of scripts may be installed in the Scripts menu.

Scripts added to the Scripts folder while Adobe Photoshop CS3 is running will not appear in the Scripts menu until the next time you launch the application.

All scripts found in the Scripts folder and sub-folders are displayed at the top level of the **File > Scripts** menu. The addition of sub-folders does not add a hierarchical organization to the Scripts menu.

Executing other scripts

The **Browse** item at the end of the **Scripts** menu (**File > Scripts > Browse**) allows you to execute scripts which are not installed in the Scripts folder. You can also use Browse to select scripts installed in the Scripts folder after the application was last launched.

Selecting **Browse** displays a file browser dialog which allows you to select a script file for execution. Only `.js` or `.jsx` files are displayed in the browse dialog. When you select a script file, it is executed the same way as an installed script.

Startup scripts

On startup, Adobe Photoshop CS3 executes all `.jsx` files that it finds in the startup folders.

- On Windows, the startup folder for user-defined scripts is:

  `C:\Program Files\Common Files\Adobe\Startup Scripts CS3\Adobe Photoshop`

- On Mac OS, the startup folder for user-defined scripts is:

  `~/Library/Application Support/Adobe/Startup Scripts CS3/Adobe Photoshop`

If a script is meant to be executed only by Adobe Photoshop CS3, it must include code such as the following:

```
if( BridgeTalk.appName == "photoshop" ) {
    //continue executing script
}
```

For additional details, see the *JavaScript Tools Guide CS3*.

Changes Since Earlier Versions

The following changes have been made to the JavaScript object model and language support in Adobe Photoshop CS3:

- Documentation Changes:
  - Documentation for ExtendScript objects (such as File and Folder objects, Script UI, and the Dollar ($) object) is now found in a separate manual entitled the *JavaScript Tools Guide CS3*.
    - On Windows, the document can be found in the following folder:

      `C:\Program Files\Adobe\Adobe Utilities\ExtendScript Toolkit 2\SDK`

    - On Mac OS, the document can be found in the following folder:

      `Applications/Utilities/Adobe Utilities/ExtendScript Toolkit 2/SDK`

  - Documentation that provides an overview of Scripting is now found in a new document entitled *Introduction to Scripting*, instead of in the *Photoshop CS3 Scripting Guide*.

- Photoshop CS3 has an Extended Version and a Standard Version. Some additional features are available in the Extended Version.

- `CountItems` provides support in scripting for the Count Tool. A new collection, `CountItems`, is defined, with a corresponding `countItems` property on `Document`, that refers to the collection `CountItems`. This feature is available only in the Extended Version.

- `ColorSamplers` have been provided for the `Document` object. A new collection, `ColorSamplers`, is defined, with a corresponding `colorSamplers` property on `Document` that refers to the collection `ColorSamplers`.

- `DICOMOpenOptions:` a new class that provides options for opening files in the DICOM format; this feature is only available in the Extended Version.

- The measurement scale feature adds a `measurementScale` property to the `Document` object, which accesses the `MeasurementScale` object. The `measurementScale` property on `Document` is read-only (i.e. you cannot create a new `MeasurementScale` object), but the properties of the `MeasurementScale` object (`pixelLength`, `logicalLength`, and `logicalUnits`) can be set directly. This feature is available only in the Extended Version.

- New JavaScript Resource that allows scripts to behave like plug-ins. Provides a way to specify a menu the script appears in, a terminology resource to allow the script to function with the Action Manager, and a way to indicate whether the scripts is enabled in the menu or not.

- New and modified methods:

  - `Application.featureEnabled()`: Allows you to determine if a feature with a given name is enabled, as related to the Extended version of Photoshop.

  - `Application.refresh`(): Pauses the script while the application refreshes.

  - `Application.openDialog():` Allows you to use the Photosho open dialog to select files.

  - `Application.putCustomOptions()`, `Application.getCustomOptions()`, `Application.eraseCustomOptions()`: Allow you to save and load your parameters

  - `Application.open():` Includes a boolean parameter `smartObject`, which indicates whether to create a smart object around the document opened.

  - `Document.duplicate()`: Provides new parameters for naming the duplciated document and setting a "merge visible layers" only option.

  - `Documents.add()`: Provides new parameters for document depth and color profile name.

  - `Document.suspendHistory()`: Allows you to provide a JavaScript string that executes as a single history state. Provides one undo and history item for multiple changes to a document.

- New properties:

  - `scriptingBuildDate` added to the `Application` object.

  - `recentFiles` added to the `Application` object.

  - `Solid`, boolean, added to `Selection` object, indicates if the bounding rectangle is solid.

- Modified enumerations:

  - `HARDMIX` has been added to the BlendMode enumeration.

  - `HARDMIXBLEND` has been added to the ColorBlendMode enumeration.

- PHOTOFILTER and EXPOSURE have been added to the LayerKind enumeration.
- DICOM has been added to the OpenDocumentType enumeration.

Bug fixes

- Fixed bounds property on Selection object.
- Fixed PathItem.duplicate() method, it now returns the PathItem.
- The Document.resizeImage command now behaves correctly for percentages. In CS2, it was off by 2 decimal places. (E.g 25% = 0.25)
- The LensBlurOptions object has been removed. All options for the Lens Blur filter now appear directly as parameters to the ArtLayer.applyLensBlur() method.
- Preferences propeties macOSTumbnail and windowsTumbnail have been revised to correct the spelling error, to macOSThumbnail and windowsThumbnail.
- Fixed CameraRAWOpenOptions.

2 JavaScript Object Reference

The objects of CameraRAW, the JavaScript type library for Adobe® Photoshop® CS3, are presented alphabetically and in tabular format in this chapter.

Object properties and methods are described in separate tables for each object. See Working with the Properties Tables and Working with the Methods Tables for information on how to use these tables.

Sample code for several object model classes is given to help illustrate the syntax as well as usage of the object class.

Working with the Properties Tables

The Properties table for an object lists the following:

- The properties you can use with the object
- The value type for each property

  When the value type is a constant or another object, the value is a hypertext link to the constant's or object's listing, as in the following Properties table sample.
- The property's input status: read-only or read-write.
- A description that explains what the property is

  Descriptions are omitted for self-explanatory properties.

Property	Value Type	What it is
`displayDialogs`	`DialogModes`	Read-write. Controls whether or not Adobe Photoshop CS3 displays dialog boxes.

Working with the Methods Tables

The Methods table for an object lists the following:

- The method name
- Parameter(s)

  When a parameter type or return value is a constant or another object, the value is a hypertext link to the constant's or object's listing. In the following Methods table sample, the parameter type `ActionDescriptor` is an object; the parameter type `DialogModes` is a constant; the return value `ActionDescriptor` is also an object.

  Appeasements can be required or optional. Optional parameters are indicated in the table by square brackets ([]). See 'Working with Method Parameters' on page 36 for information on using parameters.
- Return value type(s)
- A description, if applicable

Method	Parameter Type	Returns	What it does
executeAction (eventID [, descriptor] [, displayDialogs])	number (long) ActionDescriptor DialogModes	ActionDescriptor	Plays an ActionManager event.

Working with Method Parameters

Optional parameters are surrounded by square brackets ([ ]). In the following Methods table sample, the parameters descriptor and displayDialogs are optional and the parameter eventID is not.

Therefore, if you use the executeAction() method for the object associated with the sample Methods table above, you *must* include an eventID value in the parentheses following the method name. The eventID value must be a number, as indicated by the number (long) in the table's Parameter Type column.

If you use an optional parameter, you must separate the parameters with a comma, as indicated by the comma that precedes each optional parameter in the table.

Also, if you use an optional parameter, you must enter the values in the order they are listed in the table so that the JavaScript compiler knows which value you are entering. To skip an optional parameter, insert an extra comma to act as a placeholder.

The following sample provides values for an eventID and a displayDialog, but skips the descriptor parameter (represented by the empty value between two commas). The statement executes action #4233 and allows only error type dialog boxes to be displayed.

```
app.executeAction(4233,,error)
```

ActionDescriptor

A record of key-value pairs for actions, such as those included on the Adobe Photoshop CS3 Actions menu.

Note: The `ActionDescriptor` class is part of the Action Manager functionality. For more details on the Action Manager, see the *Photoshop CS3 Scripting Guide*.

Properties

Property	Value Type	What it is
count	number (long)	Read-only. The number of keys contained in the descriptor.
typename	string	Read-only. The class name of the referenced `actionDescriptor` object.

Methods

Method	Parameter Type	Returns	What it does
clear ()			Clears the descriptor.
erase (key)	number (long)		Erases a key from the descriptor.
fromStream (value)	string		Creates a descriptor from a stream of bytes; for reading from disk.
getBoolean (key)	number (long)	boolean	Gets the value of a key of type boolean.
getClass (key)	number (long)	number (long)	Gets the value of a key of type class.
getData (key)	number (long)	string	Gets raw byte data as a string value.
getDouble (key)	number (long)	number (double)	Gets the value of a key of type double.
getEnumerationType (key)	number (long)	number (long)	Gets the enumeration type of a key.
getEnumerationValue (key)	number (long)	number (long)	Gets the enumeration value of a key.
getInteger (key)	number (long)	number (long)	Gets the value of a key of type integer.
getKey (index)	number (long)	number (long)	Gets the ID of the *N*th key, provided by index.
getList (key)	number (long)	ActionList	Gets the value of a key of type list.

Method	Parameter Type	Returns	What it does (Continued)
getObjectType (key)	number (long)	number (long)	Gets the class ID of an object in a key of type object.
getObjectValue (key)	number (long)	ActionDescriptor	Gets the value of a key of type object.
getPath (key)	number (long)	File	Gets the value of a key of type File. **Note:** For information about the File object, see the *JavaScript Tools Guide*
getReference (key)	number (long)	ActionReference	Gets the value of a key of type ActionReference.
getString (key)	number (long)	string	Gets the value of a key of type string.
getType (key)	number (long)	DescValueType	Gets the type of a key.
getUnitDoubleType (key)	number (long)	number (long)	Gets the unit type of a key of type UnitDouble.
getUnitDoubleValue (key)	number (long)	number (double)	Gets the value of a key of type UnitDouble.
hasKey (key)	number (long)	boolean	Checks whether the descriptor contains the provided key.
isEqual (otherDesc)	ActionDescriptor	boolean	Determines whether the descriptor is the same as another descriptor.
putBoolean (key, value)	number (long) boolean		Sets the value for a key whose type is boolean.
putClass (key, value)	number (long) number (long)		Sets the value for a key whose type is class.
putData (key, value)	number (long) string		Puts raw byte data as a string value.
putDouble (key, value)	number (long) number (double)		Sets the value for a key whose type is double.
putEnumerated (key, enumType, value)	number (long) number (long) number (long)		Sets the enumeration type and value for a key.
putInteger (key, value)	number (long) number (long)		Sets the value for a key whose type is integer.

Method	Parameter Type	Returns	What it does (Continued)
putList (key, value)	number (long) ActionList		Sets the value for a key whose type is an ActionList object.
putObject (key, classID, value)	number (long) number (long) ActionDescriptor		Sets the value for a key whose type is an object, represented by an Action Descriptor.
putPath (key, value)	number (long) File		Sets the value for a key whose type is path. **Note:** For information about the File object, see the *JavaScript Tools Guide*
putReference (key, value)	number (long) ActionReference		Sets the value for a key whose type is an object reference.
putString (key, value)	number (long) string		Sets the value for a key whose type is string.
putUnitDouble (key, unitID, value)	number (long) number (long) number (double)		Sets the value for a key whose type is a unit value formatted as a double.
toStream		string	Gets the entire descriptor as a stream of bytes; for writing from disk.

ActionList

The list of commands that comprise an Action (such as an Action created using the Actions palette in the Adobe Photoshop CS3 application).

Note: The `ActionList` object is part of the Action Manager functionality. For details on using the Action Manager, see the *Photoshop CS3 Scripting Guide*.

Properties

Property	Value Type	What it is
`count`	`number (long)`	Read-only. The number of commands that comprise the action.
`typename`	`string`	Read-only. The class name of the referenced `ActionList` object.

Methods

With the exception of the `clear()` method, you use the methods of this object to either get the value of a specific type of data in the list or set (put) the value type.

Method	Parameter Type	Returns	What it does

`clear`
`()`
			Clears the list.

`getBoolean`
`(index)`
	`number (long)`	`boolean`	Gets the value of a list item of type boolean.

`getClass`
`(index)`
	`number (long)`	`number (long)`	Gets the value of a list item of type class.

`getData`
`(index)`
	`number (long)`	`string`	Gets raw byte data as a string value.

`getDouble`
`(index)`
	`number (long)`	`number (double)`	Gets the value of a list item of type double.

`getEnumerationType`
`(index)`
	`number (long)`	`number (long)`	Gets the enumeration type of a list item.

`getEnumerationValue`
`(index)`
	`number (long)`	`number (long)`	Gets the enumeration value of a list item.

`getInteger`
`(index)`
	`number (long)`	`number (long)`	Gets the value of a list item of type integer.

`getList`
`(index)`
	`number (long)`	`ActionList`	Gets the value of a list item of type list.

`getObjectType`
`(index)`
	`number (long)`	`number (long)`	Gets the class ID of a list item of type object.

Method	Parameter Type	Returns	What it does (Continued)
getObjectValue (index)	number (long)	ActionDescriptor	Gets the value of a list item of type object.
getPath (index)	number (long)	File	Gets the value of a list item of type File. **Note:** For information about the File object, see the *JavaScript Tools Guide*
getReference (index)	number (long)	ActionReference	Gets the value of a list item of type ActionReference.
getString (index)	number (long)	string	Gets the value of a list item of type string.
getType (index)	number (long)	DescValueType	Gets the type of a list item.
getUnitDoubleType (index)	number (long)	number (long)	Gets the unit value type of a list item of type Double.
getUnitDoubleValue (index)	number (long)	number (double)	Gets the unit value of a list item of type double.
putBoolean (value)	boolean		Sets the value to either true or false.
putClass (value)	number (long)		Sets the class or data type.
putData (value)	string		Puts raw byte data as a string value.
putDouble (value)	number (double)		Sets the value type as a double.
putEnumerated (enumType, value)	number (long) number (long)		Sets the value type as an enumerated, or constant, value.
putInteger (value)	number (long)		Sets the value of a list item of type integer.
putList (value)	ActionList		Sets the value of a list item of type list or array.
putObject (classID, value)	number (long) ActionDescriptor		Sets the value of a list item of type object.
putPath (value)	File		Sets the value of a list item of type path. **Note:** For information about the File object, see the *JavaScript Tools Guide*

Method	Parameter Type	Returns	What it does (Continued)
putReference (value)	ActionReference		Sets the value of a list item whose type a reference to an object created in the script.
putString (value)	string		Sets the value of a list item of type string.
putUnitDouble (classID, value)	number (long) number (double)		Sets the value of a list item of type unit value represented as a double.

ActionReference

Contains data describing a referenced Action.

Note: The `ActionReference` object is part of the Action Manager functionality. For details on using the Action Manager, see the *Photoshop CS3 Scripting Guide*.

Properties

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

Methods

Method	Parameter Type	Returns	What it does

`getContainer`
`()`
		`ActionReference`	Gets a reference contained in this reference.
			Container references provide additional pieces to the reference. This looks like another reference, but it is actually part of the same reference.

`getDesiredClass`
`()`
		`number (long)`	Gets a number representing the class of the object.

`getEnumeratedType`
`()`
		`number (long)`	Gets the enumeration type.

`getEnumeratedValue`
`()`
		`number (long)`	Gets the enumeration value.

`getForm`
`()`
		`ReferenceFormType`	Gets the form of an `ActionReference`.

`getIdentifier`
`()`
		`number (long)`	Gets the identifier value for a reference whose form is identifier.

`getIndex`
`()`
		`number (long)`	Gets the index value for a reference in a list or array.

`getName`
`()`
		`string`	Gets the name of a reference.

`getOffset`
`()`
		`number (long)`	Gets the offset of the object's index value.

`getProperty`
`()`
		`number (long)`	Gets the property ID value.

`putClass`
`(desiredClass)`
	`number (long)`		Puts a new class form and class type into the reference.

Method	Parameter Type	Returns	What it does (Continued)
putEnumerated (desiredClass, enumType, value)	number (long) number (long) number (long)		Puts an enumeration type and ID into a reference along with the desired class for the reference.
putIdentifier (desiredClass, value)	number (long) number (long)		Puts a new identifier and value into the reference..
putIndex (desiredClass, value)	number (long) number (long)		Puts a new index and value into the reference.
putName (desiredClass, value)	number (long) string		Puts a new name and value into the reference.
putOffset (desiredClass, value)	number (long) number (long)		Puts a new offset and value into the reference.
putProperty (desiredClass, value)	number (long) number (long)		Puts a new property and value into the reference.

Application

The Adobe Adobe Photoshop CS3 application object, which contains all other Adobe Photoshop CS3 objects.

Note: Because you open JavaScripts through the application itself, you do not need to use the `Application` object as part of the containment hierarchy that describes an object.

However, if you choose to include the `Application` object in your code, you must use the pre-defined global object name `app`, rather than the class name `Application`, in a script, as in the following sample:

```
var docRef = app.documents.add(800, 600, 72, "docRef", NewDocumentMode.RGB)
```

The following sample uses the `Application` object incorrectly:

```
var docRef = Application.documents.add(800, 600, 72, "docRef",
NewDocumentMode.RGB)
```

However, the most common way to add an element in your code is to omit references to the `Application` object altogether, as in the following sample:

```
var docRef = documents.add(800, 600, 72, "docRef", NewDocumentMode.RGB)
```

Properties

Property	Value Type	What it is
activeDocument	Document	Read-write. The frontmost document. (Setting this property is equivalent to clicking an open document in the Adobe Photoshop CS3 application to bring it to the front of the screen.)
backgroundColor	SolidColor	Read-write. The color mode for the document's background color.
colorSettings	String	Read-write. The name of selected color setting's set.
displayDialogs	DialogModes	Read-write. The dialog mode for the document, which indicates whether or not Adobe Photoshop CS3 displays dialogs when the script runs.
documents	Documents	Read-only. The collection of open documents.
fonts	TextFonts	Read-only. The fonts installed on this system.
foregroundColor	SolidColor	Read-write. The default foreground color (used to paint, fill, and stroke selections).
freeMemory	number (double)	Read-only. The amount of unused memory available to Adobe Photoshop CS3.
locale	string	Read-only. The language location of the application.
macintoshFileTypes	array of strings	Read-only. A list of file image types Adobe Photoshop CS3 can open.
measurementLog	MeasurementLog	The log of measurements taken.

Property	Value Type	What it is (Continued)
name	string	Read-only. The application's name.
notifiers	Notifiers	Read-only. The collection of notifiers currently configured (in the Scripts Events Manager menu in the Adobe Photoshop CS3 application).
notifiersEnabled	boolean	Read-write. Indicates whether all notifiers are enabled or disabled.
path	File	Read-only. The full path to the location of the Adobe Photoshop CS3 application.
playbackDisplayDialogs	DialogModes	Read-write. The dialog mode for playback mode, which indicates whether or not Adobe Photoshop CS3 displays dialogs in playback mode.
playbackParameters	ActionDescriptor	Read-write. The playback options, which indicate the speed at which Adobe Photoshop CS3 plays actions.
preferences	Preferences	Read-only. The application preference settings (equivalent to selecting **Edit > Preferences** in the Adobe Photoshop CS3 application in Windows or Photoshop > Preferences in Mac OS).
preferencesFolder	File	Read-only. The full path to the Preferences folder. **Note:** For information about the File object, see the *JavaScript Tools Guide*
recentFiles	array of File	Read-only. Files in the Recent Files list.
scriptingBuildDate	string	Read-only. The build date of the Scripting interface.
scriptingVersion	string	Read-only. The version of the Scripting interface.
typename	string	Read-only. The class name of the referenced app object.
version	string	Read-only. The version of Adobe Photoshop application you are running.
windowsFileTypes	array of strings	Read-only. A list of file image extensions Adobe Photoshop CS3 can open.

Methods

Method	Parameter Type	Returns	What it does
batch (inputFiles, action, from [, options])	array of File string string BatchOptions	string	Runs the batch automation routine **(similar to the Batch command, or File > Automate > Batch in the** Adobe Photoshop CS3 application). **Note:** The inputFiles parameter specifies the source for the files to be manipulated by the Batch command. **Note:** For information about the File object, see the *JavaScript Tools Guide*
beep ()			Causes a "beep" sound.
bringToFront			Makes Adobe Photoshop CS3 the active (front-most) application.
charIDToTypeID (charID)	string	number (long)	Converts from a four character code (character ID) to a runtime ID.
doAction (action, from)	string string		Plays an action from the Actions palette.
eraseCustomOptions (key)	string		Erases user object with ID value key from the Photoshop registry.
executeAction (eventID [, descriptor] [, displayDialogs])	number (long) ActionDescriptor DialogModes	ActionDescriptor	Plays an ActionManager event.
executeActionGet (reference)	ActionReference	ActionDescriptor	Obtains an ActionDescriptor.

Method	Parameter Type	Returns	What it does (Continued)

featureEnabled
(name
)
	string	boolean	Determines whether the feature specified by `name` is enabled.
			The following features are supported as values for `name`:
			`“photoshop/extended”`
			`“photoshop/standard”`
			`“photoshop/trial”`

getCustomOptions (key)	string	ActionDescriptor	Retreives user objects in the Photoshop registry for the ID with value `key`.
load (document)	File		Loads the support document from the specified location. **Note:** For information about the File object, see the *JavaScript Tools Guide*
makeContactSheet (inputFiles [, options])	array of File ContactSheetOptions	string	Creates a contact sheet from the specified files.
makePDFPresentation (inputFiles, outputFiles [, options])	array of File File PresentationOptions	string	Creates an Adobe PDF presentation file from the specified input files. **Note:** The return string contains the path to the PDF file.
makePhotoGallery (inputFolder, outputFolder [, options])	File File GalleryOptions	string	Creates a web photo gallery from the files in the specified input folder.

makePhotomerge
(inputFiles)
	array of File	string	Deprecated for Adobe Photoshop CS3. Use:
			`runphotomergeFromScript = true;`
			`$.evalFile( app.Path + "Presets/Scripts/Photomerge.jsx")photomerge.createPanorama( fileList, displayDialog );`
			Merges multiple files into one; user interaction required.

makePicturePackage (inputFiles [, options])	array of File PicturePackageOptions	string	Creates a picture package from the specified input files.

Method	Parameter Type	Returns	What it does (Continued)
open (document [, as] [, asSmartObject])	File object (open options) boolean **Note:** See individual file type open options, such as CameraRAWOpenOptions or EPSOpenOptions, etc.	Document	Opens the specified document as the optionally specified file type. Optional paramater asSmartObject (default: false) indicates whether to create a smart object around the opened document. **Note:** For information about the File object, see the *JavaScript Tools Guide*. See the Application sample script for an example of using the File object in the open method.
openDialog ()		array of File	Uses the Photoshop open dialog box to select files. Returns an array of File representing the files selected. **Note:** For information about the File object, see the *JavaScript Tools Guide*
purge (target)	PurgeTarget		Purges one or more caches.
putCustomOptions (key, customObject [, persistent])	string ActionDescriptor boolean		Saves user objects in the Photoshop registry. key provides the unique ID for your user object. customObject provides the object to save in the registry. persistent indicates whether the object should persist once the script has finished.
refresh ()			Pauses the script while the application refreshes.
stringIDToTypeID (stringID)	string	number (long)	Converts from a string ID to a runtime ID.

Method	Parameter Type	Returns	What it does (Continued)
typeIDToCharID (typeID)	number (long)	string	Converts from a runtime ID to a character ID.
typeIDToStringID (typeID)	number (long)	string	Converts from a runtime ID to a string ID.

First Sample Script

The following script invokes an alert box to display Properties important to an application such as version number, the path to the application, the amount of memory available, and the number of documents open.

When a user presses the OK button on the alert box, a second dialog opens, which asks users whether they would like the foreground and background colors set for the document presently open. If no document is open, the script opens a new document for the user.

The script (with no document open) produces a progression of three dialogs.

Application.jsx

```
//Create a Welcome message
// Use the name and version properties of the application object to
// Append the application’s name and version to the Welcome message
// use "\r" to insert a carriage return
// use the combination operator += to append info to the message
var message = "Welcome to " + app.name
message += " version " + app.version + "\r\r"

// find out where Adobe Photoshop CS3 is installed
// and add the path to the message
// add the optional parameter fsName to the path property
// to display the file system name in the most common format
message += "I’m installed in " + app.path.fsName + "\r\r"

// see how much memory Adobe Photoshop CS3 has to play with
message += "You have this much memory available for Adobe Photoshop CS3: " +
app.freeMemory + "\r\r"

// use the length property of the documents object to
// see how many documents are open
var documentsOpen = app.documents.length
message += "You currently have " + documentsOpen + " document(s) open.\r\r"

// display the message to the user
alert(message)

// answer will be true for a "Yes" answer and false for a "No" answer
var answer = confirm("Do you want me to set the foreground and background to my
favorite colors?")

// set the colors
if (answer) {
   // I don’t have a favorite color. Why did I ask you may wonder?
   app.foregroundColor.rgb.red = Math.random() * 255
   app.foregroundColor.rgb.green = Math.random() * 255
   app.foregroundColor.rgb.blue = Math.random() * 255
   app.backgroundColor.rgb.red = Math.random() * 255
   app.backgroundColor.rgb.green = Math.random() * 255
   app.backgroundColor.rgb.blue = Math.random() * 255
}

// Open a document
if (app.documents.length == 0) {

   // use the application’s path and the offset to the samples folder
   var sampleDocToOpen = File(app.path + "/Samples/Fish.psd")

   // compose a message with the name of the file
   message = "Would you like me to open a sample for you? ("
   message += sampleDocToOpen.fsName
   message += ")"

   // ask the user another question
   answer = confirm(message)

   // open the document accordingly
   if (answer) {
   open(sampleDocToOpen)
   }
}
```

Second Sample Script

The following script presents a progression of images as an Adobe PDF slide show.

PDFPresentation.jsx

```
// use all the files in the Samples folder
var inputFolder = new Folder(app.path + "/Samples/")

// see if we have something interesting
if (inputFolder != null) {

      // get all the files found in this folder that are Adobe Photoshop CS3 (.psd
format)
      var inputFiles = inputFolder.getFiles("*.psd")

      // output to the desktop
      var outputFile = File("~/Desktop/JavaScriptPresentation.pdf")

      // there are defaults but I like to set the options myself
      var options = new PresentationOptions
      options.presentation = true
      options.view = true
      options.autoAdvance = true
      options.interval = 5
      options.loop = true
      options.transition = TransitionType.RANDOM

      // create the presentation
      makePDFPresentation(inputFiles, outputFile, options)

alert("Presentation file saved to: " + outputFile.fsName)

}
```

Note: To run this code on non-English platforms, substitute the following path for the `outputFile` variable:

```
var outputFile = File("~/JavaScriptPresentation.pdf")
```

ArtLayer

An object within a document that contains the visual elements of the image (equivalent to a layer in the Adobe Photoshop CS3 application).

Note: Most likely, you will use variables to refer to `ArtLayer` objects in your script. However, if you choose not to use a variable, be aware that, because the `ArtLayer` class is also a property of the Document object, you use the property name, `artLayer`, rather than the class name, `ArtLayer`, in your code.

The following example uses correct syntax to refer to an `ArtLayer` object by name and then assign its `allLocked` property value:

```
documents(0).artLayer(“my layer”).allLocked = true
```

The following example, which uses an upper case *A* in the object name, is incorrect:

```
documents(0).ArtLayer(“my layer”).allLocked = true
```

Properties

Property	Value Type	What it is
`allLocked`	`boolean`	Read-write. Indicates whether to completely lock the layer’s contents and settings.
`blendMode`	`BlendMode`	Read-write. The layer’s blending mode.
`bounds`	`array(UnitValue)`	Read-only. An array of coordinates that describes the bounding rectangle of the layer. **Note:** For information about the UnitValue type, see the *JavaScript Tools Guide*.
`fillOpacity`	`number (double)`	Read-write. The interior opacity of the layer (between 0.0 and 100.0).
`grouped`	`boolean`	Read-write. Indicates this layer is grouped with the layer beneath it.
`isBackgroundLayer`	`boolean`	Read-write. Indicates whether the layer is a background layer or normal layer. **Note:** A document can have only one background layer.
`kind`	`LayerKind`	Read-write. Sets the layer’s kind (such as 'text layer') for an empty layer. **Note:** Valid only when the layer is empty and when `isBackgroundLayer` is `false`. See isBackgroundLayer. **Note:** You can use the `kind` property to make a background layer a normal layer; however, to make a layer a background layer, you must set `isBackgroundLayer` to `true`.

Property	Value Type	What it is (Continued)
`linkedLayers`	`array of ArtLayer or LayerSet objects`	Read-only. The layers linked to this layer. **Note:** See link.
`name`	`string`	Read-write. The layer's name.
`opacity`	`number (double)`	Read-write. The master opacity of the layer (0.0 - 100.0).
`parent`	`object (Document)`	Read-only. The object's container.
`pixelsLocked`	`boolean`	Read-write. Indicates whether the pixels in the layer's image can be edited using the paintbrush tool.
`positionLocked`	`boolean`	Read-write. Indicates whether the pixels in the layer's image can be moved within the layer.
`textItem`	`TextItem`	Read-only. The text item that is associated with the layer. **Note:** Valid only when `kind = LayerKind.TEXT`. See kind.
`transparentPixelsLocked`	`boolean`	Read-write. Indicates whether editing is confined to the opaque portions of the layer.
`typename`	`string`	Read-only. The class name of the referenced `artLayer` object.
`visible`	`boolean`	Read-write. Indicates whether the layer is visible.

Methods

Method	Parameter Type	Returns	What it does

`adjustBrightnessContrast`
`(brightness,`
`contrast)`
	`number (long)`
	`number (long)`
			Adjusts the brightness (-100 - 100) and contrast (-100 - 100).

`adjustColorBalance`
`([shadows]`
`[, midtones]`
`[, highlights]`
`[, preserveLuminosity]`
	`array of integers`
	`array of integers`
	`array of integers`
	`boolean`
			Adjusts the color balance of the layer's component channels. For `shadows`, `midtones`, and `highlights`, the array must include three values (-100 - 100), which represent cyan or red, magenta or green, and yellow or blue, when the document mode is CMYK or RGB.
			Note: See mode in the Properties table of the `Document` object.

Method	Parameter Type	Returns	What it does (Continued)

`adjustCurves`
`(curveShape)`
	`array of points (Array (Array(x, y)))`		Adjusts the tonal range of the selected channel using up to fourteen points.

`adjustLevels`
`(inputRangeStart,`
`inputRangeEnd,`
`inputRangeGamma,`
`outputRangeStart,`
`outputRangeEnd)`
	`number (long)`
	`number (long)`
	`number (double)`
	`number (long)`
	`number (long)`
			Adjusts the levels of the selected channels (`inputRangeStart`: 0 - 253; `inputRangeEnd`: (`inputRangeStart` + 2) - 255; `inputRangeGamma`: 0.10 - 9.99; `outputRangeStart`: 0 - 253; `outputRangeEnd`: (`outputRangeStart` + 2) - 255.

`applyAddNoise`
`(amount,`
`distribution,`
`monochromatic)`
	`number (double)`
	`NoiseDistribution`
	`boolean`
			Applies the Add Noise filter (`amount`: 0.1 - 400, as a percentage).

`applyAverage`
`()`
			Applies the Average filter.

`applyBlur`
`()`
			Applies the Blur filter.

`applyBlurMore`
`()`
			Applies the Blur More filter.

`applyClouds`
`()`
			Applies the Clouds filter.

`applyCustomFilter`
`(characteristics,`
`scale,`
`offset)`
	`array of twenty-five numbers (long)`
	`number (long)`
	`number (long)`
			Applies a custom filter.
			Note: Required parameter values define the filter. Refer to Adobe Photoshop CS3 Help for specific instructions.

`applyDeInterlace`
`(eliminateFields,`
`createFields)`
	`EliminateFields`
	`CreateFields`
			Applies the De-Interlace filter.

`applyDespeckle`
`()`
			Applies the Despeckle filter.

`applyDifferenceClouds`
`()`
			Applies the Difference Clouds filter.

`applyDiffuseGlow`
`(graininess,`
`glowAmount,`
`clearAmount)`
	`number (long)`
	`number (long)`
	`number (long)`
			Applies the Diffuse Glow filter (`graininess`: 0 - 10; `glowAmount`: 0 - 20; `clearAmount`: 0 - 20).

Method	Parameter Type	Returns	What it does (Continued)
applyDisplace (horizontalScale, verticalScale, displacement, undefinedareas, displacementMapFiles)	number (long) number (long) DisplacementMapType UndefinedAreas File		Applies the Displace filter using the specified horizontal and vertical scale (-999 - 999), mapping type, treatment of undistorted areas, and path to the distortion image map.
applyDustAndScratches (radius, threshold)	number (long) number (long)		Applies the Dust & Scratches filter (radius: 1 - 100; threshold: 0 - 255).
applyGaussianBlur (radius)	number (double)		Applies the Gaussian Blur filter within the specified radius (in pixels) (0.1 - 250.0).
applyGlassEffect (distortion, smoothness, scaling [, invert] [, texture] [, textureFile])	number (long) number (long) number (long) boolean TextureType File		Applies the Glass filter (distortion: 0 - 20; smoothness: 1 - 15; scaling (in percent): 50 - 200). **Note:** For information about the File object, see the *JavaScript Tools Guide*
applyHighPass (radius)	number (double)		Applies the High Pass filter within the specified radius (in pixels) (0.1 - 250.0).

Method	Parameter Type	Returns	What it does (Continued)

`applyLensBlur`
`([source]`
`[, focalDistance]`
`[, invertDepthMap]`
`[, shape]`
`[, radius]`
`[, bladeCurvature]`
`[, rotation]`
`[, brightness]`
`[, threshold]`
`[, amount]`
`[, distribution]`
`[, monochromatic]`
`)`
	`DepthMapSource`
	`number (long)`
	`boolean`
	`Geometry`
	`number (long)`
	`number (long)`
	`number (long)`
	`number (long)`
	`number (long)`
	`number (long)`
	`NoiseDistribution`
	`boolean`
			Applies the Lens Blur filter.
			`source`: the source for the depth map (default: `DepthMapSource.NONE`)
			`focalDistance` : the blur focal distance for the depth map (default: `0`).
			`invertDepthMask` : whether the depth map is inverted (default: `false`).
			`shape`: The shape of the iris (default: `Geometry.HEXAGON`)
			`radius`: The radius of the iris (default: `15`).
			`bladeCurvature`: The blade curvature of the iris (default: `0`).
			`rotation`: The rotation of the iris (default: `0`)
			`brightness`: The brightness for the specular highlights (default: `0`).
			`threshold`: The threshold for the specular highlights (default: `0`).
			`amount`: The amount of noise (default: `0`)
			`distribution`: The distribution value for the noise (default: `NoiseDistribution.UNIFORM`).
			`monochromatic`: Indicates whether the noise is monochromatic (default: `false`).

`applyLensFlare`
`(brightness,`
`flareCenter,`
`lensType)`
	`number (long)`
	`array(UnitValue)`
	`LensType`
			Applies the Lens Flare filter with the specified brightness (0 - 300, as a percentage), the x and y coordinates (unit value) of the flare center, and the lens type.
			Note: For information about the `UnitValue` type, see the *JavaScript Tools Guide*

Method	Parameter Type	Returns	What it does (Continued)
applyMaximum (radius)	number (double)		Applies the Maximum filter within the specified radius (in pixels) (1 - 100).
applyMedianNoise (radius)	number (double)		Applies the Median Noise filter within the specified radius (in pixels) (1 - 100).
applyMinimum (radius)	number (double)		Applies the Minimum filter within the specified radius (in pixels) (1 - 100).
applyMotionBlur (angle, radius)	number (long) number (double)		Applies the Motion Blur filter (angle: -360 - 360; radius: 1 - 999).
applyNTSC ()			Applies the NTSC colors filter.
applyOceanRipple (size, magnitude)	number (long) number (long)		Applies the Ocean Ripple filter in the specified size (1 - 15) and magnitude (0 - 20).
applyOffset (horizontal, vertical, undefinedAreas)	UnitValue UnitValue OffsetUndefinedAreas		Moves the layer the specified amount horizontally and vertically (min/max amounts depend on layer size), leaving an undefined area at the layer's original location. **Note:** For information about the UnitValue type, see the *JavaScript Tools Guide*
applyPinch (amount)	number (long)		Applies the Pinch filter in the specified amount (as a percentage) (-100 - 100).
applyPolarCoordinates (conversion)	PolarConversionType		Applies the Polar Coordinates filter.
applyRadialBlur (amount, blurMethod, blurQuality)	number (long) RadialBlurMethod RadialBlurQuality		Applies the Radial Blur filter in the specified amount (1 - 100) using either a spin or zoom effect and the specified quality.
applyRipple (amount, size)	number (long) RippleSize		Applies the Ripple filter in the specified amount (-999 to 999) throughout the image and in the specified size.
applySharpen ()			Applies the Sharpen filter.

Method	Parameter Type	Returns	What it does (Continued)
applySharpenEdges ()			Applies the Sharpen Edges filter.
applySharpenMore ()			Applies the Sharpen More filter.
applyShear (curve, undefinedAreas)	array of points (Array (Array(x, y))) UndefinedAreas		Applies the Shear filter (curve: 2 - 255 points).
applySmartBlur (radius, threshold, blurQuality, mode)	number (double) number (double) SmartBlurQuality SmartBlurMode		Applies the Smart Blur filter (radius: 0.1 - 100.0; threshold: 0.1 - 100.0).
applySpherize (amount, mode)	number (long) SpherizeMode		Applies the Spherize filter in the specified amount (as percentage) (-100 - 100).
applyStyle (styleName)	string		Applies the specified style to the layer. **Note:** You must use a style from the Styles list in the Layer Styles Palette.
applyTextureFill (textureFile)	File		Applies the Texture Fill filter. **Note:** For information about the File object, see the *JavaScript Tools Guide*
applyTwirl (angle)	number (long)		Applies the Twirl filter at the specified angle (-999 - 999).
applyUnSharpMask (amount, radius, threshold)	number (double) number (double) number (long)		Applies the Unsharp Mask filter (amount: 1 - 500 as percent; radius: 0.1 - 250.00; threshold: 0 - 255).
applyWave (generatorNumber, minimumWavelength, maximumWavelength, minimumAmplitude, maximumAmplitude, horizontalScale, verticalScale, waveType, undefinedAreas, randomSeed)	number (long) number (long) number (long) number (long) number (long) number (long) number (long) WaveType UndefinedAreas number (long)		Applies the Wave filter (generatorNumber**: 1 - 999;** minimumWavelength: **1 - 998;** maximumWavelength**:** **2 -** minimumWavelength + 1**;** minimumAmplitude**: 1 - 998;** maximumAmplitude**:** **2 -** minimumAmplitude + 1**;** horizontalScale**: 1 - 100,** as a percentage; verticalScale**: 1- 100**, as a percentage).

Method	Parameter Type	Returns	What it does (Continued)
applyZigZag (amount, ridges, style)	number (long) number (long) ZigZagType		Applies the Zigzag filter (amount: -100 - 100; ridges: 0 - 20).
autoContrast ()			Adjusts the contrast of the selected channels automatically.
autoLevels ()			Adjusts the levels of the selected channels using the auto levels option.
clear ()			Cuts the layer without moving it to the clipboard.
copy ([merge])	boolean		Copies the layer to the clipboard. When the optional argument is set to true, a merged copy is performed (that is, all visible layers are copied to the clipboard).
cut ()			Cuts the layer to the clipboard.
desaturate ()			Converts a color image to a grayscale image in the current color mode by assigning equal values of each component color to each pixel.
duplicate ([relativeObject] [, insertionLocation])	object (ArtLayer or LayerSet) ElementPlacement	ArtLayer or LayerSet	Creates a duplicate of the object on the screen.
equalize ()			Redistributes the brightness values of pixels in an image to more evenly represent the entire range of brightness levels within the image.
invert ()			Inverts the colors in the layer by converting the brightness value of each pixel in the channels to the inverse value on the 256-step color-values scale.
link (with)	object (ArtLayer or LayerSet)		Links the layer with the specified layer.

Method	Parameter Type	Returns	What it does (Continued)
merge ()		ArtLayer	Merges the layer down, removing the layer from the document; returns a reference to the art layer that this layer is merged into.
mixChannels (outputChannels [, monochrome])	array of array of numbers (double) boolean		Modifies a targeted (output) color channel using a mix of the existing color channels in the image. (`outputChannels` = An array of channel specifications. For each component channel, specify a list of adjustment values (-200 - 200) followed by a 'constant' value (-200 - 200).) **Note:** When `monochrome = true`, the maximum number of channel value specifications is 1. **Note:** Valid only when `docRef.mode = DocumentMode.RGB` or `docRef.mode = DocumentMode.CMYK`. **Note:** RGB arrays must include four doubles. CMYK arrays must include five doubles.
move (relativeObject, insertionLocation)	object (ArtLayer or LayerSet) ElementPlacement		Moves the layer relative to the object specified in parameters. **Note:** For art layers, only the constant values `ElementPlacement.PLACEBEFORE` and `ElementPlacement.PLACEAFTER` are valid. For layer sets, only the constant values `ElementPlacement.PLACEBEFORE` and `ElementPlacement.INSIDE` are valid.

Method	Parameter Type	Returns	What it does (Continued)
photoFilter ([fillColor] [, density] [, preserveLuminosity])	SolidColor number (long) boolean		Adjust the layer's color balance and temperature as if a color filter had been applied (density: 1 - 100, as a percentage).
posterize (levels)	number (long)		Specifies the number of tonal levels (2 - 255) for each channel and then maps pixels to the closest matching level.
rasterize (target)	RasterizeType		Converts the targeted contents in the layer into a flat, raster image.
remove ()			Deletes the object.
resize ([horizontal] [, vertical] [, anchor])	number (double) number (double) AnchorPosition		Resizes the layer to the specified dimensions (as a percentage of its current size) and places it in the specified position.
rotate (angle [, anchor])	number (double) AnchorPosition		Rotates rotates the layer around the specified anchor point (default: AnchorPosition.MIDDLECENTER).
selectiveColor (selectionMethod [, reds] [, yellows] [, greens] [, cyans] [, blues] [, magentas] [, whites] [, neutrals] [, blacks])	AdjustmentReference array of numbers (long) array of numbers (long) array of numbers (long) array of numbers (long) array of numbers (long) array of numbers (long) array of numbers (long) array of numbers (long) array of numbers (long)		Modifies the amount of a process color in a specified primary color without affecting the other primary colors. **Note:** Each color array must have four components.
shadowHighlight ([shadowAmount] [, shadowWidth] [, shadowRadius] [, highlightAmount] [, highlightWidth] [, highlightRadius] [, colorCorrection] [, midtoneContrast] [, blackClip] [, whiteClip])	number (long) number (long) number (long) number (long) number (long) number (long) number (long) number (long) number (double) number (double)		Adjusts the range of tones in the image's shadows and highlights (shadowAmount: 0 - 100 as percent; shadowWidth: 0 - 100 as percent; shadowRadius: 0 - 2500 in pixels; highlightAmount: 0 - 100 as percent; highlightWidth: 0 - 100 as percent; highlightRadius: 0 - 2500 in pixels; colorCorrection: -100 - 100; midtoneContrast: -100 - 100; blackClip: 0.000 - 50.000; whiteClip: 0.000 - 50.000).

Method	Parameter Type	Returns	What it does (Continued)
threshold (level)	number (long)		Converts grayscale or color images to high-contrast, B/W images by converting pixels lighter than the specified threshold to white and pixels darker than the threshold to black (level: 1 - 255).
translate ([deltaX] [, deltaY])	UnitValue UnitValue		Moves the layer the specified amount (in pixels) relative to its current position. **Note:** For information about the `UnitValue` type, see the *JavaScript Tools Guide*
unlink ()			Unlinks the layer.

Sample Script

The following script creates art layers to display a duck and a sand dune in an overlying checkerboard pattern. A multi-layered collage displays.

ArtLayer.jsx

```
// Save the current preferences
var startRulerUnits = app.preferences.rulerUnits
var startTypeUnits = app.preferences.typeUnits
var startDisplayDialogs = app.displayDialogs

// Set Adobe Photoshop CS3 to use pixels and display no dialogs
app.preferences.rulerUnits = Units.PIXELS
app.preferences.typeUnits = TypeUnits.PIXELS
app.displayDialogs = DialogModes.NO

//Close all the open documents
 while (app.documents.length) {
   app.activeDocument.close()
}

// Create a new document to merge all the samples into
var mergedDoc = app.documents.add(1000, 1000, 72, "Merged Samples",
NewDocumentMode.RGB, DocumentFill.TRANSPARENT, 1)

// Use the path to the application and append the samples folder
var samplesFolder = Folder(app.path + "/Samples/")

//Get all the files in the folder
var fileList = samplesFolder.getFiles()

// open each file
for (var i = 0; i < fileList.length; i++) {
   // The fileList is folders and files so open only files
   if (fileList[i] instanceof File) {
         open(fileList[i])

         // use the document name for the layer name in the merged document
         var docName = app.activeDocument.name

         // flatten the document so we get everything and then copy
         app.activeDocument.flatten()
         app.activeDocument.selection.selectAll()
         app.activeDocument.selection.copy()

         // don’t save anything we did
         app.activeDocument.close(SaveOptions.DONOTSAVECHANGES)

         // make a random selection on the document to paste into
         // by dividing the document up in 4 quadrants and pasting
         // into one of them by selecting that area
         var topLeftH = Math.floor(Math.random() * 2)
         var topLeftV = Math.floor(Math.random() * 2)
         var docH = app.activeDocument.width.value / 2
         var docV = app.activeDocument.height.value / 2
         var selRegion = Array(Array(topLeftH * docH, topLeftV * docV),
            Array(topLeftH * docH + docH, topLeftV * docV),
            Array(topLeftH * docH + docH, topLeftV * docV + docV),
            Array(topLeftH * docH, topLeftV * docV + docV),
            Array(topLeftH * docH, topLeftV * docV))
         app.activeDocument.selection.select(selRegion)
         app.activeDocument.paste()

         // change the layer name and opacity
         app.activeDocument.activeLayer.name = docName
         app.activeDocument.activeLayer.fillOpacity = 50
   }
}

// sort the layers by name
for (var x = 0; x < app.activeDocument.layers.length; x++) {
   for (var y = 0; y < app.activeDocument.layers.length - 1 - x; y++) {
         // Compare in a non-case sensitive way
         var doc1 = app.activeDocument.layers[y].name
         var doc2 = app.activeDocument.layers[y + 1].name
         if (doc1.toUpperCase() > doc2.toUpperCase()) {
            app.activeDocument.layers[y].move(app.activeDocument.layers[y+1],
               ElementPlacement.PLACEAFTER)
      }
   }
}

// Reset the application preferences
app.preferences.rulerUnits = startRulerUnits
app.preferences.typeUnits = startTypeUnits
app.displayDialogs = startDisplayDialogs
```

ArtLayers

The collection of `artLayer` objects in the document.

Note: Because the `ArtLayers` class is a property of the Document object, you use the property name, `artLayers`, rather than the class name, `ArtLayers`, in your code. For example:

```
var layerRef = docRef.artLayers.add()
```

The following sample uses the `ArtLayers` object incorrectly:

```
var layerRef = docRef.ArtLayers.add()
```

Properties

Property	Value Type	What it is
`length`	`number (long)`	Read-only. The number of elements in the `artLayers` collection.
`parent`	`object (Document)`	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the referenced `artLayers` object.

Methods

Method	Parameter Type	Returns	What it does

`index`
`(itemKey)`
	`number`	`ArtLayer`	Gets an element from the `artLayers` collection.

`add`
`()`
		`ArtLayer`	Creates a new `artLayer` in the document.

`getByName`
`(name)`
	`string`	`ArtLayer`	Get the first element in the `artLayers` collection with the provided name.

`removeAll`
`()`
		`Nothing`	Removes all elements from the `artLayers` collection.

BatchOptions

Options to specify when running a Batch command.

Note: You specify the batch source folder as the `inputFiles` parameter of the `batch()` method, which is a method of the `Application` class. See 'batch' on page 47. JavaScript supports only folders as sources for batch commands.

Properties

Property	Value type	What it is
`destination`	`BatchDestinationType`	Read-write. The type of destination for the processed files (default: `BatchDestinationType.NODESTINATION`).
`destinationFolder`	`File`	Read-write. The folder location for the processed files. **Note:** Valid only when `destination = BatchDestinationType.FOLDER`. See destination. **Note:** For information about the `File` object, see the *JavaScript Tools Guide*
`errorFile`	`File`	Read-write. The file in which to log errors encountered. **Note:** To display errors on the screen (and stop batch processing when errors occur) leave blank. **Note:** For information about the `File` object, see the *JavaScript Tools Guide*
`fileNaming`	`array of FileNamingType options`	Read-write. A list of file naming options (maximum: 6). **Note:** Valid only when `destination = BatchDestinationType.FOLDER`. See destination.
`macintoshCompatible`	`boolean`	Read-write. Indicates whether to make the final file names Macintosh compatible (default: `true`). **Note:** Valid only when `destination = BatchDestinationType.FOLDER`. See destination.
`overrideOpen`	`boolean`	Read-write. Indicates whether to override action open commands (default: `false`).

Property	Value type	What it is (Continued)
`overrideSave`	`boolean`	Read-write. Indicates whether to override save as action steps with the specified destination (default: `false`). **Note:** Valid only when `destination = BatchDestinationType.FOLDER` or `destination = BatchDestinationType.SAVEANDCLOSE`. See destination.
`startingSerial`	`number (long)`	Read-write. The starting serial number to use in naming files (default: `1`). **Note:** Valid only when `destination = BatchDestinationType.FOLDER`. See destination.
`suppressOpen`	`boolean`	Read-write. Indicates whether to suppress the file open options dialogs (default: `false`).
`suppressProfile`	`boolean`	Read-write. Indicates whether to suppress the color profile warnings (default: `false`).
`typename`	`string`	Read-only. The class name of the referenced `batchOptions` object.
`unixCompatible`	`boolean`	Read-write. Indicates whether to make the final file name Unix compatible (default: `true`). **Note:** Valid only when `destination = BatchDestinationType.FOLDER`. See destination.
`windowsCompatible`	`boolean`	Read-write. Indicates whether to make the final file names Windows compatible (default: `true`). **Note:** Valid only when `destination = BatchDestinationType.FOLDER`. See destination.

BitmapConversionOptions

Options to be specified when converting an image to Bitmap mode.

Note: Convert color images to grayscale before converting the image to bitmap mode. See 'desaturate' on page 60 (in the Methods table of the ArtLayer object).

Properties

Property	Value Type	What it is
angle	number (double)	Read-write. The angle (in degrees) at which to orient individual dots (-180 - 180). See shape. **Note:** Valid only when method = BitmapConversionType.HALFTONESCREEN. See method.
frequency	number (double)	Read-write. The number of printer dots (per inch) to use (1.0 - 999.99). **Note:** Valid only when method = BitmapConversionType.HALFTONESCREEN. See method.
method	BitmapConversionType	Read-write. The conversion method to use (default: BitmapConversionType.DIFFUSIONDITHER).
patternName	string	Read-write. The name of the pattern to use. For information about pre-installed valid patterns, see Adobe Photoshop CS3 Help on the bitmap conversion command, or view the options availabe in the Custom Color drop down box after choosing the bitmap conversion command. **Note:** Valid only when method = BitmapConversionType.CUSTOMPATTERN. See method.
resolution	number (double)	Read-write. The output resolution in pixels per inch (default: 72.0).
shape	BitmapHalfToneType	Read-write. The dot shape to use. **Note:** Valid only when method = BitmapConversionType.HALFTONESCREEN. See method.
typename	string	Read-only. The class name of the referenced bitmapConversionOptions object.

BMPSaveOptions

Options that can be specified when saving a document in BMP format.

Properties

Property	Value Type	What it is
alphaChannels	boolean	Read-write. Indicates whether to save the alpha channels.
depth	BMPDepthType	Read-write. The number of bits per channel.
flipRowOrder	boolean	Read-write. Indicates whether to write the image from top to bottom (default: false). **Note:** Available only when osType = OperatingSystem.WINDOWS. See osType.
osType	OperatingSystem	Read-write. The target OS. (default: OperatingSystem.WINDOWS).
rleCompression	boolean	Read-write. Indicates whether to use RLE compression. **Note:** Available only when osType = OperatingSystem.WINDOWS. See osType.
typename	string	Read-only. The class name of the referenced BMPSaveOptions object.

CameraRAWOpenOptions

Options that can be specified when opening a document in Camera RAW format.

Properties

Property	Value type	What it is
bitsPerChannel	BitsPerChannelType	Read-write. The number of bits per channel.
blueHue	number (long)	Read-write. The blue hue of the shot (-100 - 100).
blueSaturation	number (long)	Read-write. The blue saturation of the shot (-100 - 100).
brightness	number (long)	Read-write. The brightness of the shot (0 - 150).
chromaticAberrationBY	number (long)	Read-write. The chromatic aberration B/Y of the shot (-100 - 100).
chromaticAberrationRC	number (long)	Read-write. The chromatic aberration R/C of the shot (-100 - 100).
colorNoiseReduction	number (long)	Read-write. The color noise reduction of the shot (0 - 100).
colorSpace	ColorSpaceType	Read-write. The colorspace for the image.
contrast	number (long)	Read-write. The contrast of the shot (-50 - 100).
exposure	number (double)	Read-write. The exposure of the shot (4.0 - 4.0).
greenHue	number (long)	Read-write. The green hue of the shot (-100 - 100).
greenSaturation	number (long)	Read-write. The green saturation of the shot (-100 - 100).
luminanceSmoothing	number (long)	Read-write. The luminance smoothing of the shot (0 - 100).
redHue	number (long)	Read-write. The red hue of the shot (-100 - 100).
redSaturation	number (long)	Read-write. The red saturation of the shot (-100 - 100).
resolution	number (double)	Read-write. The resolution of the document in pixels per inch (1 - 999).
saturation	number (long)	Read-write. The saturation of the shot (-100 - 100).
settings	CameraRAWSettingsType	Read-write. The global settings for all Camera RAW options. Default: CameraRAWSettingsType.CAMERA.
shadows	number (long)	Read-write. The shadows of the shot (0 - 100).

Property	Value type	What it is (Continued)
shadowTint	number (long)	Read-write. The shadow tint of the shot (-100 - 100).
sharpness	number (long)	Read-write. The sharpness of the shot (0 - 100).
size	CameraRAWSize	Read-write. The size of the new document.
temperature	number (long)	Read-write. The temperature of the shot (2000 - 50000).
tint	number (long)	Read-write. The tint of the shot (-150 - 150).
typename	string	Read-only. The class name of the referenced cameraRAWOpenOptions object.
vignettingAmount	number (long)	Read-write. The vignetting amount of the shot (-100 - 100).
vignettingMidpoint	number (long)	Read-write. The vignetting mid point of the shot (-100 - 100).
whiteBalance	WhiteBalanceType	Read-write. The white balance options for the image.

Channel

Object that stores information about a color element in the image, analogous to a plate in the printing process that applies a single color. The document's color mode determines the number of default channels; for example, an RGB document has four default channels:

- A composite channel: RGB
- Three component channels: red, green, blue

A channel can also be an alpha channel, which stores selections as masks, or a spot channel, which stores spot colors.

Note: Most likely, you will use variables to refer to `Channel` objects in your script. However, if you choose not to use a variable, be aware that, because the `Channel` class is a property of the Document object, you use the property name, `channel`, rather than the class name, `Channel`, in your code.

The following example uses correct syntax to refer to a `Channel` object by name and then assign its `opacity` property value:

```
documents(0).channel("my channelr").opacity = 22
```

The following example, which uses an upper case *C* in the object name, is incorrect:

```
documents(0).Channel("my channelr").opacity = 22
```

Properties

Property	Value Type	What it is
`color`	`SolidColor`	Read-write. The color of the channel. **Note:** Not valid when `type = ChannelType.COMPONENT`.
`histogram`	`array of 256 numbers (long)`	Read-only. A histogram of the color of the channel. **Note:** Not valid when `type = ChannelType.COMPONENT`. For component channel histogram values, use the histogram property of the Document object instead.
`kind`	`ChannelType`	Read-write. The channel type.
`name`	`string`	Read-write. The channel's name.
`opacity`	`number (double)`	Read-write. The opacity to use for alpha channels or the solidity to use for spot channels (0 - 100). **Note:** Valid only when `type = ChannelType.MASKEDAREA` or `type = ChannelType.SELECTEDAREA`.
`parent`	`object (Document)`	Read-only. The object's container.

Property	Value Type	What it is (Continued)
typename	string	Read-only. The class name of the referenced channel object.
visible	boolean	Read-write. Indicates whether the channel is visible.

Methods

Method	Parameter Type	Returns	What it does
duplicate ([targetDocument])	Document	Channel	Duplicates the channel.
merge ()			Merges a spot channel into the component channels.
remove ()			Deletes the channel.

Channels

The collection of `channel` objects in the document. See `Channel`.

Note: Because the `Channels` class is also a property of the Document object, you use the property name, `channels`, rather than the class name, `Channels`, in your code. For example:

```
var channelRef = docRef.channels.add()
```

The following sample uses the `Channels` object incorrectly:

```
var channelRef = docRef.Channels.add()
```

Properties

Property	Value Type	What it is
`length`	`number (long)`	Read-only. The number of elements in the `channels` collection.
`parent`	`object (Document)`	Read-only. The `channels` object's container.
`typename`	`string`	Read-only. The class name of the referenced `channels` object.

Methods

Method	Parameter Type	Returns	What it does

`index`
`(itemKey)`
	`number`	`Channel`	Gets an element from the `channels` collection.

`add`
`()`
		`Channel`	Creates a new `channel` object.

`getByName`
`(name)`
	`string`	`Channel`	Get the first element in the `channels` collection with the provided name.

`removeAll`
`()`
			Removes all `channel` objects from the `channels` collection.

Sample Script

The following script opens a file if one is not already open, and then writes a histogram report (histogram.log) for the channels in the active document.

Note: This script contains a switch construction that uses a `break` statement. The `break` statement requires an ending semicolon (;), as in the following sample:

```
break;
```

Histogram.jsx

```
// Save the current preferences
var startRulerUnits = app.preferences.rulerUnits
var startTypeUnits = app.preferences.typeUnits
var startDisplayDialogs = app.displayDialogs
```

```
// Set Adobe Photoshop CS3 to use pixels and display no dialogs
app.preferences.rulerUnits = Units.PIXELS
app.preferences.typeUnits = TypeUnits.PIXELS
app.displayDialogs = DialogModes.NO

// if there are no documents open then try to open a sample file
if (app.documents.length == 0) {
   open(File(app.path + "/Samples/Fish.psd"))
}

// get a reference to the working document
var docRef = app.activeDocument

// create the output file
// first figure out which kind of line feeds we need
if ($.os.search(/windows/i) != -1) {
   fileLineFeed = "windows"
} else {
   fileLineFeed = "macintosh"
}

// create the output file accordingly
fileOut = new File("~/Desktop/Histogram.log")
fileOut.lineFeed = fileLineFeed
fileOut.open("w", "TEXT", "????")

// write out a header
fileOut.write("Histogram report for " + docRef.name)

// find out how many pixels I have
var totalCount = docRef.width.value * docRef.height.value

// more info to the out file
fileOut.write(" with a total pixel count of " + totalCount + "\n")

// channel indexer
var channelIndex = 0

// remember which channels are currently active
var activeChannels = app.activeDocument.activeChannels

// document histogram only works in these modes
if (docRef.mode == DocumentMode.RGB ||
    docRef.mode == DocumentMode.INDEXEDCOLOR ||
    docRef.mode == DocumentMode.CMYK) {

   // activate the main channels so we can get the documents histogram
   TurnOnDocumentHistogramChannels(docRef)

   // Output the documents histogram
   OutputHistogram(docRef.histogram, "Luminosity", fileOut)
}

// local reference to work from
var myChannels = docRef.channels

// loop through each channel and output the histogram
for (var channelIndex = 0; channelIndex < myChannels.length; channelIndex++) {
   // the channel has to be visible to get a histogram
   myChannels[channelIndex].visible= true

   // turn off all the other channels
   for (var secondaryIndex = 0; secondaryIndex < myChannels.length;
             secondaryIndex++) {
          if (channelIndex != secondaryIndex) {
             myChannels[secondaryIndex].visible= false
      }
   }

   // Use the function to dump the histogram
   OutputHistogram(myChannels[channelIndex].histogram,
          myChannels[channelIndex].name, fileOut)
}

// close down the output file
fileOut.close()
alert("Histogram file saved to: " + fileOut.fsName)

// reset the active channels
docRef.activeChannels = activeChannels

// Reset the application preferences
app.preferences.rulerUnits = startRulerUnits
app.preferences.typeUnits = startTypeUnits
app.displayDialogs = startDisplayDialogs


// Utility function that takes a histogram and name
// and dumps to the output file
function OutputHistogram(inHistogram, inHistogramName, inOutFile) {

   // find ouch which count has the largest number
   // I scale everything to this number for the output
   var largestCount = 0

   // a simple indexer I can reuse
   var histogramIndex = 0

   // see how many samples we have total
   var histogramCount = 0

   // search through all and find the largest single item
   for (histogramIndex = 0; histogramIndex < inHistogram.length;
             histogramIndex++) {
          histogramCount += inHistogram[histogramIndex]
          if (inHistogram[histogramIndex] > largestCount)
             largestCount = inHistogram[histogramIndex]
   }

   // These should match
   if (histogramCount != totalCount) {
          alert("Something bad is happening!")
   }

   // see how much each "X" is going to count as
   var pixelsPerX = largestCount / 100

   // output this data to the file
   inOutFile.write("One X = " + pixelsPerX + " pixels.\n")

   // output the name of this histogram
   inOutFile.write(inHistogramName + "\n")

   // loop through all the items and output in the following format
   // 001
   // 002
   for (histogramIndex = 0; histogramIndex < inHistogram.length;
             histogramIndex++) {

          // I need an extra "0" for this line item to keep everything in line
          if (histogramIndex < 10)
             inOutFile.write("0")

          // I need an extra "0" for this line item to keep everything in line
          if (histogramIndex < 100)
             inOutFile.write("0")

          // output the index to file
          inOutFile.write(histogramIndex)

          // some spacing to make it look nice
          inOutFile.write(" ")

          // figure out how many X’s I need
          var outputX = inHistogram[histogramIndex] / largestCount * 100

          // output the X’s
          for (var a = 0; a < outputX; a++)
             inOutFile.write("X")

          inOutFile.write("\n")
   }

   inOutFile.write("\n")
}

// Function to active all the channels according to the documents mode
// Takes a document reference for input
function TurnOnDocumentHistogramChannels(inDocument) {

   // see how many channels we need to activate
   var visibleChannelCount = 0

   // based on the mode of the document
   switch (inDocument.mode) {

          case DocumentMode.BITMAP:
          case DocumentMode.GRAYSCALE:
          case DocumentMode.INDEXEDCOLOR:
             visibleChannelCount = 1
             break;

          case DocumentMode.DUOTONE:
             visibleChannelCount = 2
             break;

          case DocumentMode.RGB:
          case DocumentMode.LAB:
             visibleChannelCount = 3
             break;

          case DocumentMode.CMYK:
             visibleChannelCount = 4
             break;

          case DocumentMode.DUOTONE:
             visibleChannelCount = 4
             break;

          case DocumentMode.MULTICHANNEL:
          default:
             visibleChannelCount = inDocument.channels.length + 1
             break;
   }

   // now get the channels to activate into a local array
   var aChannelArray = new Array()

   // index for the active channels array
   var aChannelIndex = 0

   for(var channelIndex = 0; channelIndex < inDocument.channels.length;
             channelIndex++) {
          if (channelIndex < visibleChannelCount) {
             aChannelArray[aChannelIndex++] = inDocument.channels[channelIndex]
       }
   }

   // now activate them
   inDocument.activeChannels = aChannelArray

}
```

CMYKColor

The definition of a CMYK color.

Properties

Property	Value Type	What it is
black	number (double)	Read-write. The black color value (as percent) (0.0 - 100.0).
cyan	number (double)	Read-write. The cyan color value (as percent) (0.0 - 100.0).
magenta	number (double)	Read-write. The magenta color value (as percent) (0.0 - 100.0).
typename	string	Read-only. The class name of the referenced CMYKColor object.
yellow	number (double)	Read-write. The yellow color value (as percent) (0.0 - 100.0).

ColorSampler

A color sampler for the document.

Note: For additional information about color samplers, see Adobe Photoshop CS3 help on the Color SamplerTool.

Properties

Property	Value type	What it is
color	SolidColor	Read-only. The color of the color sampler.
position	array of UnitValue	Read-only. The position of the color sampler in the document. The array (x,y) represents the horizontal and vertical location of the count item. **Note:** For information about the `UnitValue` object, see the *JavaScript Tools Guide*
parent	object (Document)	Read-only. The `ColorSampler` object's container.
typename	string	Read-only. The class name of the referenced `ColorSampler` object.

Methods

Method	Parameter type	Returns	What it does
move (position)	array of UnitValue		Moves the color sampler to a new location in the document. The `position` parameter (x,y) represents the new horizontal and vertical locations, respectively, of the moved color sampler. **Note:** For information about the `UnitValue` object, see the *JavaScript Tools Guide*
remove ()			Deletes the `ColorSampler` object.

ColorSamplers

The collection of `ColorSampler` objects in the document. See `ColorSampler`.

Note: Because the `ColorSampler` class is also a property of the Document object, you use the property name, `colorSamplers`, rather than the class name, `ColorSamplers`, in your code. For example:

```
docRef.colorSamplers.removeAll()
```

The following sample uses the `ColorSamplers` object incorrectly:

```
docRef.ColorSamplers.removeAll()
```

Properties

Property	Value Type	What it is
`length`	`number (long)`	Read-only. The number of elements in the `ColorSamplers` collection.
`parent`	`object (Document)`	Read-only. The `ColorSamplers` object's container.
`typename`	`string`	Read-only. The class name of the referenced `ColorSamplers` object.

Methods

Method	Parameter Type	Returns	What it does

`index`
`(itemKey)`
	`number`	`ColorSampler`	Gets an element from the `ColorSamplers` collection.

`add`
`(position)`
	`array of UnitValue`	`ColorSampler`	Creates a new `ColorSampler` object.
			The `position` parameter (x,y) represents the new horizontal and vertical locations, respectively, of the moved color sampler.
			Note: For information about the `UnitValue` object, see the *JavaScript Tools Guide*

`getByName`
`(name)`
	`string`	`ColorSampler`	Get the first element in the `ColorSamplers` collection with the provided name.

`removeAll`
`()`
			Removes all `ColorSampler` objects from the `ColorSamplers` collection.

ContactSheetOptions

Options that can be specified for a contact sheet.

Properties

Property	Value Type	What it is
acrossFirst	boolean	Read-write. Indicates whether to place the images horizontally (left to right, then top to bottom) first (default: `true`).
bestFit	boolean	Read-write. Indicates whether to rotate images for the best fit (default: `false`).
caption	boolean	Read-write. Indicates whether to use the filename as a caption for the image (default: `true`).
columnCount	number (long)	Read-write. The number of columns to include (1 - 100; default: `5`).
flatten	boolean	Read-write. Indicates whether to flatten all layers in the final document (default: `true`).
font	GalleryFontType	Read-write. The font used for the caption (default: `GalleryFontType.ARIAL`).
fontSize	number (long)	Read-write. The font size to use for the caption (default: `12`).
height	number (long)	Read-write. The height (in pixels) of the resulting document (100 - 2900; default: `720`).
horizontal	number (long)	Read-write. The horizontal spacing (in pixels) between images (0 - 29000; default: `1`).
mode	NewDocumentMode	Read-write. The document color mode (default: `NewDocumentMode.RGB`).
resolution	number (double)	Read-write. The resolution of the document in pixels per inch (35 - 1200; default: `72.0`).
rowCount	number (long)	Read-write. The number of rows to use (1 - 100; default: `6`).
typename	string	Read-only. The class name of the referenced `contactSheetOptions` object.
useAutoSpacing	boolean	Read-write. Indicates whether to auto space the images (default: `true`).

Property	Value Type	What it is (Continued)
`vertical`	`number (long)`	Read-write. The vertical spacing (in pixels) between images (0 - 29000; default: `1`). **Note:** Valid only when `useAutoSpacing = false`.
`width`	`number (long)`	Read-write. The width (in pixels) of the resulting document (100 - 2900; default: `576`).

CountItem

A counted item in the document. Also see the method autoCount, defined on Document.

Note: `CountItems` is available in the Extended Version only.

For additional information about count items, see Adobe Photoshop CS3 help on the Count Tool.

Properties

Property	Value type	What it is
`position`	`array of UnitValue`	Read-only. The position of the count item in the document. **Note:** For information about the `UnitValue` object, see the *JavaScript Tools Guide*
`parent`	`object (Document)`	Read-only. The `CountItem` object's container.
`typename`	`string`	Read-only. The class name of the referenced `CountItem` object.

Methods

Method	Parameter type	Returns	What it does
`remove` `()`			Deletes the `CountItem` object.

CountItems

The collection of `CountItem` objects in the document. See `CountItem`.

Note: `CountItem` is available in the Extended Version only.

Because the `CountItems` class is also a property of the Document object, you use the property name, `countItems`, rather than the class name, `CountItems`, in your code. For example:

```
docRef.countItems.removeAll()
```

The following sample uses the `CountItems` object incorrectly:

```
docRef.CountItems.removeAll()
```

Properties

Property	Value Type	What it is
`length`	`number (long)`	Read-only. The number of elements in the `CountItems` collection.
`parent`	`object (Document)`	Read-only. The `CountItems` object's container.
`typename`	`string`	Read-only. The class name of the referenced `CountItems` object.

Methods

Method	Parameter Type	Returns	What it does
`index` `(itemKey)`	`number`	`CountItem`	Gets an element from the `CountItems` collection.
`add` `(position)`	`array of UnitValue`	`CountItem`	Creates a new `CountItem` object. Parameter `position` (x,y) represents the horizontal and vertical positions, respectively, of the `CountItem` object. **Note:** For information about the `UnitValue` object, see the *JavaScript Tools Guide*
`getByName` `(name)`	`string`	`CountItem`	Get the first element in the `CountItems` collection with the provided name.
`removeAll` `()`			Removes all `CountItem` objects from the `CountItems` collection .

DCS1_SaveOptions

Options that can be specified when saving a CMYK document in DCS1 format.

Properties

Property	Value Type	What it is
dCS	DCSType	Read-write. (default: DCSType.COLORCOMPOSITE).
embedColorProfile	boolean	Read-write. Indicates whether to embed the color profile in the document
encoding	SaveEncoding	Read-write. The type of encoding to use for document (default: SaveEncoding.BINARY).
halftoneScreen	boolean	Read-write. Indicates whether to include halftone screen (default: false).
interpolation	boolean	Read-write. Indicates whether to use image interpolation (default: false)
preview	Preview	Read-write. The type of preview (default: Preview.MACOSEIGHTBIT).
transferFunction	boolean	Read-write. Indicates whether to include the Transfer functions to compensate for dot gain between the image and film (default: false).
typename	string	Read-only. The class name of the referenced DCS1_SaveOptions object.
vectorData	boolean	Read-write. Indicates whether to include vector data. **Note:** Valid only if the document includes vector data (un-rasterized text).

DCS2_SaveOptions

Options that can be specified when saving a CMYK document in DCS2 format.

Properties

Property	Value Type	What it is
dCS	DCSType	Read-write. The type of composite file to create (default: DCSType.NOCOMPOSITE).
embedColorProfile	boolean	Read-write. Indicates whether to embed the color profile in the document.
encoding	SaveEncoding	Read-write. The type of encoding to use (default: SaveEncoding.BINARY).
halftoneScreen	boolean	Read-write. Indicates whether to include the halftone screen (default: false).
interpolation	boolean	Read-write. Indicates whether to use image interpolation (default: false).
multiFileDCS	boolean	Read-write. Indicates whether to save color channels as multiple files or a single file (default: false).
preview	Preview	Read-write. The preview type (default: Preview.MACOSEIGHTBIT).
spotColors	boolean	Read-write. Indicates whether to save spot colors.
transferFunction	boolean	Read-write. Indicates whether to include the Transfer functions to compensate for dot gain between the image and film (default: false).
typename	string	Read-only. The class name of the referenced DCS2_SaveOptions object.
vectorData	boolean	Read-write. Indicates whether to include vector data. **Note:** Valid only if the document includes vector data (un-rasterized text).

DICOMOpenOptions

Options that can be specified when opening an DICOM format document.

Note: `DICOMOpenOptions` is available in the Extended Version only.

Properties

Property	**Value Type**	**What it is**
`anonymize`	`boolean`	Read-write. Indicates whether to make the patient information anonymous.
`columns`	`number (long)`	Read-write. Number of columns in n-up configuration.
`reverse`	`boolean`	Read-write. Indicates whether to reverse (invert) the image.
`rows`	`number (long)`	Read-write. The number of rows in n-up configuration.
`showOverlays`	`boolean`	Read-write. Indicates whether to show overlays.
`typename`	`string`	Read-only. The class name of the referenced `DICOMOpenOptions` object.
`windowLevel`	`number (long)`	Read-write. The contrast of the image in Houndsfield units.
`windowWidth`	`number (long)`	Read-write. The brightness of the image in Houndsfield units.

Document

The active containment object for layers and all other objects in the script; the basic canvas for the file.

Note: In Adobe Photoshop CS3, a document can also be referred to as an image or a canvas.

- The term *image* refers to the entire document and its contents. You can trim or crop an image. You resize an image using the `resizeImage()` method.
- The term *canvas* refers to the space in which the document sits on the screen. You can rotate or flip the canvas. You resize the canvas using the `resizeCanvas()` method.

Note: Most likely, you will use variables to refer to `Document` objects in your script. However, if you choose not to use a variable, be aware that, because the `Document` class is a property of the Application object, you use the property name, `document`, rather than the class name, `Document`, in your code.

The following example uses correct syntax to refer to a `Document` object by name and then assign its `colorProfileType` property value:

```
document("my document").colorProfileType = ColorProfile.CUSTOM
```

The following example, which uses an upper case *D* in the object name, is incorrect:

```
Document("my document").colorProfileType = ColorProfile.CUSTOM
```

Properties

Property	Value Type	What it is
`activeChannels`	array of Channel objects	Read-write. The selected channels.
`activeHistoryBrushSource`	HistoryState	Read-write. The history state to use with the history brush.
`activeHistoryState`	HistoryState	Read-write. The selected `HistoryState` object.
`activeLayer`	object (ArtLayer or LayerSet)	Read-write. The selected layer.
`artLayers`	ArtLayers	Read-only. The `artLayers` collection.
`backgroundLayer`	ArtLayer	Read-only. The background layer of the document.
`bitsPerChannel`	BitsPerChannelType	Read-write. The number of bits per channel.
`channels`	Channels	Read-only. The `channels` collection.
`colorProfileName`	string	Read-write. The name of the color profile. **Note:** Valid only when `colorProfileType = ColorProfile.CUSTOM` or `colorProfileType = ColorProfile.WORKING`. See colorProfileType.
`colorProfileType`	ColorProfile	Read-write. The type of color model that defines the document's working space.

Property	Value Type	What it is (Continued)
colorSamplers	ColorSamplers	Read-only. The current color samplers associated with this document.
componentChannels	array of Channel objects	Read-only. A list of the component color channels.
countItems	CountItems	Read-only. The current count items. **Note:** For additional information about count items, see Adobe Photoshop CS3 help on the Count Tool.
fullName	File	Read-only. The full path name of the document. **Note:** For information about the File object, see the *JavaScript Tools Guide*
height	UnitValue	Read-only. The height of the document (unit value). **Note:** For information about the UnitValue object, see the *JavaScript Tools Guide*

histogram	array of 256 numbers (long)	Read-only. A histogram showing the number of pixels at each color intensity level for the composite channel.
		Note: Valid only when
		mode = DocumentMode.RGB;
		mode = DocumentMode.CMYK; or
		mode = DocumentMode.INDEXEDCOLOR.
		See mode.

historyStates	HistoryStates	Read-only. The HistoryStates collection.
info	DocumentInfo	Read-only. Metadata about the document.
layerComps	LayerComps	Read-only. The LayerComps collection.
layers	Layers	Read-only. The Layers collection.
layerSets	LayerSets	Read-only. The LayerSets collection.
managed	boolean	Read-only. Indicates whether the document a is workgroup document.
measurementScale	MeasurementScale	Read-only. The measurement scale for the document. **Note:** The measurement scale feature is available in the Extended version only.
mode	DocumentMode	Read-only. The color profile.
name	string	Read-only. The document's name.

Property	Value Type	What it is (Continued)
`parent`	`Application`	Read-only. The `Document` object's container.
`path`	`File`	Read-only. The path to the document. **Note:** For information about the `File` object, see the *JavaScript Tools Guide*
`pathItems`	`PathItems`	Read-only. The `PathItems` collection.
`pixelAspectRatio`	`number (double)`	Read-write. The (custom) pixel aspect ratio to use (0.100 - 10.000).
`quickMaskMode`	`boolean`	Read-write. Indicates whether the document is in Quick Mask mode.
`resolution`	`number (double)`	Read-only. The document's resolution (in pixels per inch).
`saved`	`boolean`	Read-only. Indicates whether the document has been saved since the last change.
`selection`	`Selection`	Read-only. The selected area of the document.
`typename`	`string`	Read-only. The class name of the `Document` object.
`width`	`UnitValue`	Read-only. The width of the document (unit value). **Note:** For information about the UnitValue type, see the *JavaScript Tools Guide*.
`xmpMetadata`	`xmpMetadata`	Read-only. Camera RAW settings for the image. **Note:** Valid only for documents opened in Camera RAW format.

Methods

Method	Parameter Type	Returns	What it does
autoCount (channel, threshold)	Channel number (long)		Counts the number of objects in a document. Creates a CountItem object for each object counted. **Note:** The autoCount feature is available in the Extended Version only. For additional information about how to set up objects to count, please see the Count Tool in the Adobe Photoshop CS3 Help
changeMode (destinationMode [, options])	ChangeMode (BitmapConversionOptions or IndexedConversionOptions)		Changes the color profile.
close ([saving])	SaveOptions		Closes the document. If any changes have been made, the script presents an alert with three options: save, do not save, prompt to save. The optional parameter specifies a selection in the alert box (default: SaveOptions. PROMPTTOSAVECHANGES).
convertProfile (destinationProfile, intent [, blackPointCompensation] [, dither])	string Intent boolean boolean		Changes the color profile. **Note:** The destinationProfile parameter must be either a string that names the color mode or Working RGB, Working CMYK, Working Gray, Lab Color (meaning one of the working color spaces or Lab color).

Method	Parameter Type	Returns	What it does (Continued)
crop (bounds [, angle] [, width] [, height])	array(UnitValue) number (double) UnitValue UnitValue		Crops the document. The first parameter is an array of four coordinates that mark the portion remaining after cropping, in the following order: left, top, right, bottom. **Note:** For information about the `UnitValue` type, see the *JavaScript Tools Guide*.
duplicate ([name] [, mergeLayersOnly])	string boolean	Document	Creates a duplicate of the `document` object. The optional parameter `name` provides the name for the duplicated document. The optional parameter `mergeLayersOnly` indicates whether to only duplicate merged layers.
exportDocument (exportIn [, exportAs] [, options])	File ExportType ExportOptionsIllustrator or ExportOptionsSaveForWeb		Exports the document. **Note:** For information about the `File` object, see the *JavaScript Tools Guide*
flatten ()			Flattens all layers.
flipCanvas (direction)	Direction		Flips the image within the canvas in the specified direction.
importAnnotations (file)	File		Imports annotations into the document. **Note:** For information about the `File` object, see the *JavaScript Tools Guide*
mergeVisibleLayers ()			Flattens all visible layers in the document.

Method	Parameter Type	Returns	What it does (Continued)
paste ([intoSelection])	boolean	ArtLayer	Pastes the contents of the clipboard into the document. If the optional argument is set to `true` and a selection is active, the contents are pasted into the selection.
print ([postScriptEncoding] [, sourceSpace] [, printSpace] [, intent] [blackPointCompensation])	PrintEncoding SourceSpaceType string Intent boolean		Prints the document. **Note:** `printSpace` specifies the color space for the printer. Valid values are `nothing` (that is, the same as the source); or `Working RGB`, `Working CMYK`, `Working Gray`, `Lab Color` (meaning one of the working color spaces or Lab color); or a string specifying a specific colorspace (default: *`nothing`*).
rasterizeAllLayers ()			Rasterizes all layers.
recordMeasurements ([source] [, dataPoints])	MeasurementSource array of strings		Record measurements of document.
resizeCanvas ([width] [, height] [, anchor])	UnitValue UnitValue AnchorPosition		Changes the size of the canvas to display more or less of the image but does not change the image size. See resizeImage. **Note:** For information about the `UnitValue` object, see the *JavaScript Tools Guide*
resizeImage ([width] [, height] [, resolution] [, resampleMethod])	UnitValue UnitValue number (double) ResampleMethod		Changes the size of the image. **Note:** For information about the `UnitValue` object, see the *JavaScript Tools Guide*

Method	Parameter Type	Returns	What it does (Continued)
revealAll ()			Expands the document to show clipped sections.
rotateCanvas (angle)	number (double)		Rotates the canvas (including the image) in clockwise direction.
save ()			Saves the document.
saveAs (saveIn [, options] [, asCopy] [, extensionType])	File *format*SaveOptions object* boolean Extension * Examples: BMPSaveOptions DCS2_SaveOptions JPEGSaveOptions TiffSaveOptions etc.		Saves the document with specified save options. **Note:** For information about the File object, see the *JavaScript Tools Guide*
splitChannels ()		array of Document objects	Splits the document channels into separate images.
suspendHistory (historyString javaScriptString)	string string		Provides a single entry in history states for the entire script provided by javaScriptString. Allows a single undo for all actions taken in the script. The historyString parameter provides the string to use for the history state. The javaScriptString parameter provides a string of JavaScript code to excute while history is suspended.
trap (width)	number (long)		Applies trapping to a CMYK document. **Note:** Valid only when docRef.mode = DocumentMode.CMYK. See mode .

Method	Parameter Type	Returns	What it does (Continued)

`trim`
`([type]`
`[, top]`
`[, left]`
`[, bottom]`
`[, right])`
	`TrimType`
	`boolean`
	`boolean`
	`boolean`
	`boolean`
			Trims the transparent area around the image on the specified sides of the canvas.
			Note: Default is `true` for all boolean values.

Sample Script

The following script creates a document that contains two images (a flower and a duck) obtained from the Adobe Photoshop CS3 Samples folder and employs the following steps:

- Determines which image is larger.
- Resizes the smaller image to match the larger image.
- Creates a merged document twice as high as either image in order to hold both images.
- Selects part of the document to and pastes the flower into the selection.
- Inverts the selection and pastes the duckinto the lower part of the document.
- Positions the flower over the duck.

Document.jsx

```
// Save the current preferences
var startRulerUnits = app.preferences.rulerUnits
var startTypeUnits = app.preferences.typeUnits
var startDisplayDialogs = app.displayDialogs

// Set Adobe Photoshop CS3 to use pixels and display no dialogs
app.preferences.rulerUnits = Units.PIXELS
app.preferences.typeUnits = TypeUnits.PIXELS
app.displayDialogs = DialogModes.NO

// first close all the open documents
while (app.documents.length) {
   app.activeDocument.close()
}

// Open the sunflower and duck files from the samples folder
var flowerDoc = open(File(app.path + "/Samples/Sunflower.psd"))
var duckDoc = open(File(app.path + "/Samples/Ducky.tif"))

// Find out which document is larger
// Resize the smaller document the to the larger document’s size
// The resize requires the document be the active/front document
if ((flowerDoc.width.value * flowerDoc.height.value) > (duckDoc.width.value *
duckDoc.height.value)) {
   app.activeDocument = duckDoc
   duckDoc.resize(flowerDoc.width, flowerDoc.height)
} else {
   app.activeDocument = flowerDoc
   flowerDoc.resizeImage(duckDoc.width, duckDoc.height)
}

// Create a new document twice as high as two files
```

```
var mergedDoc = app.documents.add(duckDoc.width, duckDoc.height * 2,
duckDoc.resolution, "FlowerOverDuck")

// Copy the flower to the top; make it the active document so we can manipulate it
app.activeDocument = flowerDoc
flowerDoc.activeLayer.copy()

//Pastethe flower to the merged document, making the merged document active
app.activeDocument = mergedDoc

// Select a square area at the top of the new document
var selRegion = Array(Array(0, 0),
                   Array(mergedDoc.width.value, 0),
                   Array(mergedDoc.width.value, mergedDoc.height.value / 2),
                   Array(0, mergedDoc.height.value / 2),
                   Array(0, 0))
// Create the selection
mergedDoc.selection.select(selRegion)

//Paste in the flower
mergedDoc.paste()

// do the same thing for the duck
app.activeDocument = duckDoc
duckDoc.activeLayer.copy()

app.activeDocument = mergedDoc
mergedDoc.selection.select(selRegion)

// Inverting the selection so the bottom of the document is now selected
mergedDoc.selection.invert()

// Paste the duck
mergedDoc.paste()

// get rid of our originals without modifying them
duckDoc.close(SaveOptions.DONOTSAVECHANGES)
flowerDoc.close(SaveOptions.DONOTSAVECHANGES)

// Reset the application preferences
app.preferences.rulerUnits = startRulerUnits
app.preferences.typeUnits = startTypeUnits
app.displayDialogs = startDisplayDialogs
```

DocumentInfo

Metadata about a `document` object. These values can be set by selecting File > File Info in the Adobe Photoshop CS3 application.

Note: Since the `DocumentInfo` class is also a property of the Document object, you use the property name `info`, rather than the class name `DocumentInfo`, in a script, as in the following sample, which sets the `author`, `caption`, and `copyrighted` properties:

```
var docRef = open(fileList[i])
// set the file info
docRef.info.author = "Mr. Adobe programmer"
docRef.info.caption = "Adobe Photo shoot"
docRef.info.copyrighted = CopyrightedType.COPYRIGHTEDWORK
```

The following sample uses the `DocumentInfo` object incorrectly:

```
docRef.DocumentInfo.author = "Mr. Adobe programmer"
docRef.DocumentInfo.caption = "Adobe Photo shoot"
docRef.DocumentInfo.copyrighted = CopyrightedType.COPYRIGHTEDWORK
```

Properties

Property	Value Type	What it is
author	string	Read-write.
authorPosition	string	Read-write.
caption	string	Read-write.
captionWriter	string	Read-write.
category	string	Read-write.
city	string	Read-write.
copyrighted	CopyrightedType	Read-write. The copyrighted status.
copyrightNotice	string	Read-write.
country	string	Read-write.
creationDate	string	Read-write.
credit	string	Read-write.
exif	array of arrays: Array(Array (tag, tag data)), ...)	Read-only. Camera data that includes camera settings used when the image was taken. Sample array values are: tag = "camera"; tag value = "Cannon".
headline	string	Read-write.
instructions	string	Read-write.
jobName	string	Read-write.
keywords	array of strings	Read-write. A list of keywords that can identify the document or its contents.

Property	Value Type	What it is (Continued)
ownerUrl	string	Read-write.
parent	object (Document)	Read-only. The info object's container.
provinceState	string	Read-write.
source	string	Read-write.
supplementalCategories	array of strings	Read-write.
title	string	Read-write.
transmissionReference	string	Read-write.
typename	string	Read-only. The class name of the referenced info object.
urgency	Urgency	Read-write.

Sample Script

The following script sets document info (metadata) for all of the files in a specified folder and then saves the modified files as low-quality JPEG images in a new folder without changing the originals.

- Ask the user to specify the folder that contains the original files and the output folder for the JPEG images, and then check that the folders exist.
- Open each file and use the documentInfo object properties to tag it with the following metadata:
  - author: Mr. Adobe programmer
  - caption: Adobe Photo shoot
  - captionWriter: Mr. Adobe programmer
  - city: San Jose
  - copyrightNotice: Copyright (c) Adobe programmer Photography
  - copyrighted status: Copyrighted Work
  - country: USA
  - state: CA
- Save the new documents in JPEG format with a low quality setting.

DocumentInfo.jsx

```
// Save the current preferences
var startDisplayDialogs = app.displayDialogs

// Set Adobe Photoshop CS3 to use pixels and display no dialogs
app.displayDialogs = DialogModes.NO

// ask the user for the input folder
var inputFolder = Folder.selectDialog("Select a folder to tag")

// ask the user for the output folder
var outputFolder = Folder.selectDialog("Select a folder for the output files")
```

```
// see if we got something interesting from the dialog
if (inputFolder != null && outputFolder != null) {

   // get all the files found in this folder
   var fileList = inputFolder.getFiles()

   // save the outputs in JPEG
   var jpegOptions = new JPEGSaveOptions()

   // set the jpeg quality really low so the files are small
   jpegOptions.quality = 1

   // open each one in turn
   for (var i = 0; i < fileList.length; i++) {

          // The fileList includes both folders and files so open only files
          if (fileList[i] instanceof File && fileList[i].hidden == false) {

             // get a reference to the new document
             var docRef = open(fileList[i])

             // tag all of the documents with photo shoot information
             docRef.info.author = "Mr. Adobe programmer"
             docRef.info.caption = "Adobe Photo shoot"
             docRef.info.captionWriter = "Mr. Adobe programmer"
             docRef.info.city = "San Jose"
             docRef.info.copyrightNotice = "Copyright (c) Adobe programmer
                Photography"
             docRef.info.copyrighted = CopyrightedType.COPYRIGHTEDWORK
             docRef.info.country = "USA"
             docRef.info.provinceState = "CA"

             // change the date to a Adobe Photoshop CS3 date format
             // "YYYYMMDD"
             var theDate = new Date()

             // the year is from 1900 ????
             var theYear = (theDate.getYear() + 1900).toString()

             // convert the month from 0..12 to 00..12
             var theMonth = theDate.getMonth().toString()

             if (theDate.getMonth() < 10) {
                theMonth = "0" + theMonth
          }

             // convert the day from 0..31 to 00.31
             var theDay = theDate.getDate().toString()

             if (theDate.getDate() < 10) {
                theDay = "0" + theDay
          }

             // stick them all together
             docRef.info.creationDate = theYear + theMonth + theDay

             // flatten because we are saving to JPEG
             docRef.flatten()

             // go to 8 bit because we are saving to JPEG
```

```
                docRef.bitsPerChannel = BitsPerChannelType.EIGHT

                // save and close
                docRef.saveAs(new File(outputFolder + "/Output" + i + ".jpg"),
                    jpegOptions)

                // don’t modify the original
                docRef.close(SaveOptions.DONOTSAVECHANGES)
        }
    }
}

// Reset the application preferences
app.displayDialogs = startDisplayDialogs
```

Documents

The collection of open `document` objects. See `Document` for information on the `document` object.

Note: Because the `Documents` class is a property of the Application object, you use the property name, `documents`, rather than the class name, `Documents`, in your code, as in the following example:

```
documents.add(800, 500, 72, "myDocument", NewDocumentMode.RGB)
```

The following example, which uses an upper case *D* in the object name, is incorrect:

```
Documents.add(800, 500, 72, "myDocument", NewDocumentMode.RGB)
```

Properties

Property	Value Type	What it is
`length`	`number (long)`	Read-only. The number of elements in the `documents` collection.
`parent`	`object (Application)`	Read-only. The `documents` objects' container.
`typename`	`string`	Read-only. The class name of the referenced `documents` object.

Methods

Method	Parameter Type	Returns	What it does

`index`
`(itemKey)`
	`number`	`Document`	Gets an element from the `documents` collection.

`add`
`([width]`
`[, height]`
`[, resolution]`
`[, name]`
`[, mode]`
`[, initialFill]`
`[,pixelAspectRatio]`
`[, bitsPerChannel]`
`[,colorProfileName])`
	`UnitValue`
	`UnitValue`
	`number (double)`
	`string`
	`NewDocumentMode`
	`DocumentFill`
	`number (double)`
	`BitsPerChannelType`
	`string`
		`Document`	Adds a `document` object.
			`pixelAspectRatio`: range from 0.100 - 10.00. Default 1.0 for a square aspect ratio.
			`bitsPerChannelType` has a default value of `BitsPerChannelType.EIGHT`.
			Note: For information about the `UnitValue` type, see the *JavaScript Tools Guide*.

`getByName`
`(name)`
	`string`	`Document`	Gets the first element in the `documents` collection with the provided name

EPSOpenOptions

Options that can be specified when opening an EPS format document.

Properties

Property	Value Type	What it is
antiAlias	boolean	Read-write. Indicates whether to use antialias.
constrainProportions	boolean	Read-write. Indicates whether to constrain the proportions of the image.
height	UnitValue	Read-write. The height of the image (unit value). **Note:** For information about the UnitValue type, see the *JavaScript Tools Guide*
mode	OpenDocumentMode	Read-write. The color profile to use as the document mode.
resolution	number (double)	Read-write. The resolution of the document in pixels per inch.
typename	string	Read-only. The class name of the referenced EPSOpenOptions object.
width	UnitValue	Read-write. The width of the image (unit value). **Note:** For information about the UnitValue type, see the *JavaScript Tools Guide*

EPSSaveOptions

Options that can be specified when saving a document in EPS format.

Properties

Property	Value Type	What it is
embedColorProfile	boolean	Read-write. Indicates whether to embed the color profile in this document.
encoding	SaveEncoding	Read-write. The type of encoding to use (default: SaveEncoding.BINARY).
halftoneScreen	boolean	Read-write. Indicates whether to include the halftone screen (default: false).
interpolation	boolean	Read-write. Indicates whether to use image interpolation (default: false).
preview	Preview	Read-write. The preview type.
psColorManagement	boolean	Read-write. Indicates whether to use Postscript color management (default: false).
transferFunction	boolean	Read-write. Indicates whether to include the Transfer functions to compensate for dot gain between the image and film (default: false).
transparentWhites	boolean	Read-write. Indicates whether to display white areas as transparent. **Note:** Valid only when *document*.mode = DocumentMode.BITMAP. See 'mode' on page 90 (in the Properties table of the document object) or 'changeMode' on page 92 (in the Methods table of the document object).
typename	string	Read-only. The class name of the referenced EPSSaveOptions object.
vectorData	boolean	Read-write. Indicates whether to include vector data. **Note:** Valid only if the document includes vector data (text).

ExportOptionsIllustrator

Options that can be specified when exporting a PathItem object to an Adobe Illustrator® file.

Properties

Property	Value Type	What it is
path	IllustratorPathType	Read-write. The type of path to export (default: IllustratorPathType.DOCUMENTBOUNDS).
pathName	string	Read-write. The name of the path to export. **Note:** Valid only when path = IllustratorPathType.NAMEDPATH. See path.
typename	string	Read-only. The class name of the referenced exportOptionsIllustrator object.

ExportOptionsSaveForWeb

Options that can be specified when optimizing a document for the web or devices.

Properties

Property	Value type	What it is
blur	number (double)	Read-write. Applies blur to the image to reduce artifacts (default: 0.0).
colorReduction	ColorReductionType	Read-write. The color reduction algorithm (default: ColorReductionType.SELECTIVE).
colors	number (long)	Read-write. The number of colors in the palette (default: 256).
dither	Dither	Read-write. The type of dither (default: Dither.DIFFUSION).
ditherAmount	number (long)	Read-write. The amount of dither (default: 100). **Note:** Valid only when dither = Dither.DIFFUSION. See dither.
format	SaveDocumentType	Read-write. The file format to use (default: SaveDocumentType.COMPUSERVEGIF). **Note:** For this property, only COMPUSERVEGIF, JPEG, PNG-8, PNG-24, and BMP are supported.
includeProfile	boolean	Read-write. Indicates whether to include the document's embedded color profile (default: false).
interlaced	boolean	Read-write. Indicates whether to download in multiple passes; progressive (default: false).
lossy	number (long)	Read-write. The amount of lossiness allowed (default: 0).
matteColor	RGBColor	Read-write. The colors to blend transparent pixels against.
optimized	boolean	Read-write. Indicates whether to create smaller but less compatible files (default: true). **Note:** Valid only when format = SaveDocumentType.JPEG. See format.

Property	Value type	What it is (Continued)
PNG8	boolean	Read-write. Indicates the number of bits; `true` = 8, `false` = 24 (default: `true`). **Note:** Valid only when `format = SaveDocumentType.PNG`. See format.
quality	number (long)	Read-write. The quality of the produced image (0 - 100 as percentage; default: `60`).
transparency	boolean	Read-write. Indication of transparent areas of the image should be included in the saved image(default: `true`).
transparencyAmount	number (long)	Read-write. The amont of transparency dither (default: 100). **Note:** Valid only if `transparency = true`. See transparency.
transparencyDither	Dither	Read-write. The transparency dither algorithm (default: `transparencyDither = Dither.NONE`).
typename	string	Read-only. The class name of the referenced `ExportOptionsSaveForWeb` object.
webSnap	number (long)	Read-write. The tolerance amount within which to snap close colors to web palette colors (default: `0`).

GalleryBannerOptions

Options that define the bannerOptions property of the galleryOptions object. See 'GalleryOptions' on page 112.

Tip: You can preserve default values for many galleryBannerOptions properties by setting the galleryOptions property preserveAllMetadata to true or by choosing **File > Automate > Web Photo Gallery**, and then choosing **Preserve all metadata on the Options area of the Web Photo Gallery dialo**g.

Properties

Property	Value Type	What it is
contactInfo	string	Read-write. The web photo gallery contact info.
date	string	Read-write. The web photo gallery date (default: current date).
font	GalleryFontType	Read-write. The font setting for the banner text (default: GalleryFontType.ARIAL).
fontSize	number (long)	Read-write. The font size for the banner text (1 - 7; default: 3).
photographer	string	Read-write. The web photo gallery photographer.
siteName	string	Read-write. The web photo gallery site name (default: Adobe Web Photo Gallery).
typename	string	Read-only. The class name of the referenced galleryBannerOptions object.

GalleryCustomColorOptions

Options that define the `customColorOptions` property of the `galleryOptions` object. See 'GalleryOptions' on page 112.

Tip: You can preserve default values for many `galleryCustomColorOptions` properties by setting the `galleryOptions` property `preserveAllMetadata` to `true` or by choosing **File > Automate > Web Photo Gallery**, and then choosing **Preserve all metadata on the Options area of the Web Photo Gallery dial**og.

Properties

Property	Value Type	What it is
`activeLinkColor`	`RGBColor`	Read-write. The color to use to indicate an active link.
`backgroundColor`	`RGBColor`	Read-write. The background color.
`bannerColor`	`RGBColor`	Read-write. The banner color.
`linkColor`	`RGBColor`	Read-write. The color to use to indicate a link.
`textColor`	`RGBColor`	Read-write. The text color.
`typename`	`string`	Read-only. The class name of the referenced `galleryCustomColorOptions` object.
`visitedLinkColor`	`RGBColor`	Read-write. The color to use to indicate a visited link.

GalleryImagesOptions

Options that define the imagesOptions property of the galleryOptions object. See 'GalleryOptions' on page 112.

Tip: You can preserve default values for many galleryImagesOptions properties by setting the galleryOptions property preserveAllMetadata to true or by choosing **File > Automate > Web Photo Gallery**, and then choosing **Preserve all metadata on the Options area of the Web Photo Gallery dialo**g.

Properties

Property	Value Type	What it is
border	number (long)	Read-write. The size (in pixels) of the border that separates images (0 - 99; default: 0).
caption	boolean	Read-write. Indication of whether to generate image captions (default: false).
dimension	number (long)	Read-write. The resized image dimensions in pixels (default: 350). **Note:** Valid only when resizeImages = true. See resizeImages.
font	GalleryFontType	Read-write. The font to use for image captions (default: GalleryFontType.ARIAL).
fontSize	number (long)	Read-write. The font size for image captions (1 - 7; default: 3). **Note:** Valid only when caption = true. See caption.
imageQuality	number (long)	Read-write. The quality setting for a JPEG image (0 - 12; default: 5).
includeCopyright	boolean	Read-write. Indication of whether to include copyright information in captions (default: false). **Note:** Valid only when caption = true. See caption.
includeCredits	boolean	Read-write. Indication of whether to include the credits in image captions (default: false). **Note:** Valid only when caption = true. See caption.

Property	Value Type	What it is (Continued)
includeFilename	boolean	Read-write. Indication of whether to include the file name in image captions (default: true). **Note:** Valid only when caption = true. See caption.
includeTitle	boolean	Read-write. Indication of whether to include the title in image captions (default: false). **Note:** Valid only when caption = true. See caption.
numericLinks	boolean	Read-write. Indication of whether to add numeric links (default: true).
resizeConstraint	GalleryConstrainType	Read-write. The image dimensions to constrain in the gallery image (default: GalleryConstrainType.CONSTRAINBOTH). **Note:** Valid only when resizeImages = true. See resizeImages.
resizeImages	boolean	Read-write. Indication of whether to automatically resize images for placement on the gallery pages (default: true).
typename	string	Read-only. The class name of the referenced galleryImagesOptions object.

GalleryOptions

Options that can be specified for a Web photo gallery.

Tip: You can preserve default values for many `galleryOptions` properties by choosing **File > Automate > Web Photo Gallery**, and then choosing **Preserve all metadata on the Options area of the Web Photo Gallery dialog.**

Properties

Property	Value Type	What it is
addSizeAttributes	boolean	Read-write. Indicates whether width and height attributes for images will be added (default: `true`).
bannerOptions	GalleryBannerOptions	Read-write. The options related to banner settings.
customColorOptions	GalleryCustomColorOptions	Read-write. The options related to custom color settings.
emailAddress	string	Read-write. The email address to show on the web page.
imagesOptions	GalleryImagesOptions	Read-write. The options related to images settings.
includeSubFolders	boolean	Read-write. Indication of whether to include all files found in sub folders of the input folder (default: `true`).
layoutStyle	string	Read-write. The style to use for laying out the web page (default: `Centered Frame 1 - Basic`).
preserveAllMetadata	boolean	Read-write. Indicates whether to save metadata (default: `false`).
securityOptions	GallerySecurityOptions	Read-write. The options related to security settings.
thumbnailOptions	GalleryThumbnailOptions	Read-write. The options related to thumbnail image settings.
typename	string	Read-only. The class name of the referenced `galleryOptions` object.
useShortExtension	boolean	Read-write. Indicates whether the short web page extension `.htm` or number (long) web page extension `.html` will be used (default: `true`).
useUTF8Encoding	boolean	Read-write. Indicates whether the web page should use UTF-8 encoding (default: `false`).

GallerySecurityOptions

Options that define the securityOptions property of the galleryOptions object. See 'GalleryOptions' on page 112.

Tip: You can preserve default values for many gallerySecurityOptions properties by setting the galleryOptions property preserveAllMetadata to true or by choosing **File > Automate > Web Photo Gallery**, and then choosing **Preserve all metadata on the Options area of the Web Photo Gallery dia**log.

Properties

Property	Value Type	What it is
content	GallerySecurityType	Read-write. The web photo gallery security content (default: GallerySecurityType.NONE).
font	GalleryFontType	Read-write. The web photo gallery security font (default: GalleryFontType.ARIAL).
fontSize	number (long)	Read-write. The web photo gallery security font size (1 - 72; default: 3).
opacity	number (long)	Read-write. The web page security opacity as a percent (default: 100).
text	string	Read-write. The web photo gallery security custom text.
textColor	RGBColor	Read-write. The web page security text color.
textPosition	GallerySecurityTextPositionType	Read-write. The web photo gallery security text position (default: GallerySecurityTextPositionType.CENTERED).
textRotate	GallerySecurityTextRotateType	Read-write. The web photo gallery security text orientation to use (default: GallerySecurityTextRotateType.ZERO).
typename	string	Read-only. The class name of the referenced gallerySecurityOptions object.

GalleryThumbnailOptions

Options that define the `thumbnailOptions` property of the `galleryOptions` object. See 'GalleryOptions' on page 112.

Tip: You can preserve default values for many `galleryThumbnailOptions` properties by setting the `galleryOptions` property `preserveAllMetadata` to `true` or by choosing **File > Automate > Web Photo Gallery**, and then choosing **Preserve all metadata on the Options area of the Web Photo Gallery dialo**g.

Properties

Property	Value Type	What it is
border	number (long)	Read-write. The amount of border pixels you want around your thumbnail images (0 - 99; default: `0`).
caption	boolean	Read-write. Indicates whether there is a caption (default: `false`).
columnCount	number (long)	Read-write. The number of columns on the page (default: `5`).
dimension	number (long)	Read-write. The web photo gallery thumbnail dimension in pixels (default: `75).`
font	GalleryFontType	Read-write. The web photo gallery font (default: `GalleryFontType.ARIAL`).
fontSize	number (long)	Read-write. The font size for thumbnail images text (1 - 7; default: `3`).
includeCopyright	boolean	Read-write. Indication of whether to include copyright information for thumbnails (default: `false`).
includeCredits	boolean	Read-write. Indication of whether to include credits for thumbnails (default: `false`).
includeFilename	boolean	Read-write. Indication of whether to include file names for thumbnails (default: `false`).
includeTitle	boolean	Read-write. Indication of whether to include titles for thumbnails (default: `false`).
rowCount	number (long)	Read-write. The number of rows on the page (default: `3`).
size	GalleryThumbSizeType	Read-write. The thumbnail image size (default: `GalleryThumbSizeType.MEDIUM`).
typename	string	Read-only. The class name of the referenced `GalleryThumbnailOptions` object.

GIFSaveOptions

Options that can be specified when saving a document in GIF format.

Properties

Property	**Value Type**	**What it is**
`colors`	`number (long)`	Read-write. The number of palette colors. **Note:** Valid only when `palette = Palette.LOCALADAPTIVE`; `palette = Palette.LOCALPERCEPTUAL`; `palette = Palette.LOCALSELECTIVE`; `palette = Palette.MACOSPALETTE`; `palette = Palette.UNIFORM`; `palette = Palette.WEBPALETTE`; or `palette = Palette.WINDOWSPALETTE` . See palette.
`dither`	`Dither`	Read-write. The dither type.
`ditherAmount`	`number (long)`	Read-write. The amount of dither. (1 - 100; default: `75`). **Note:** Valid only when `dither = Dither.DIFFUSION`. See dither.
`forced`	`ForcedColors`	Read-write. The type of colors to force into the color palette.
`interlaced`	`boolean`	Read-write. Indicates whether rows should be interlaced (default: `false`).
`matte`	`MatteType`	Read-write. The color to use to fill anti-aliased edges adjacent to transparent areas of the image (default: `MatteType.WHITE`). **Note:** When `transparency = false`, the matte color is applied to transparent areas. See transparency.
`palette`	`Palette`	Read-write. The type of palette to use (default: `Palette.LOCALSELECTIVE`).
`preserveExactColors`	`boolean`	Read-write. Indicates whether to protect colors in the image that contain entries in the color table from being dithered. **Note:** Valid only when `dither = Dither.DIFFUSION`. See dither.

Property	Value Type	What it is (Continued)
`transparency`	`boolean`	Read-write. Indicates whether to preserve transparent areas of the image during conversion to GIF format.
`typename`	`string`	Read-only. The class name of the referenced `GIFSaveOptions` object.

GrayColor

Options for defining a gray color.

Properties

Property	Value Type	What it is
gray	number (double)	Read-write. The gray value (0.0 - 100.0; default: 0.0).
typename	string	Read-only. The class name of the referenced grayColor object.

HistoryState

A version of the document stored automatically (and added to the `HistoryStates` collection), which preserves the document's state, each time the document is changed. See HistoryStates for information about the `HistoryStates` collection.

Note: Because the `HistoryState` class is also a property of the Document object, you use the property name, `historyState`, rather than the class name, `HistoryState`, in your code.

The following example uses correct syntax to refer to a `HistoryState` object named `AddLayerMask` and then assign its `snapshot` property value:

```
documents(0).historyState("AddLayerMask").snapshot = true
```

The following example, which uses an upper case *A* in the object name, is incorrect:

```
documents(0).HistoryState("AddLayerMask").snapshot = true
```

Properties

Property	Value Type	What it is
`name`	`string`	Read-only. The `HistoryState` object's name.
`parent`	`object (Document)`	Read-only. The `HistoryState` object's container.
`snapshot`	`boolean`	Read-only. Indicates whether the history state is a snapshot.
`typename`	`string`	Read-only. The class name of the referenced `HistoryState` object.

HistoryStates

The collection of `HistoryState` objects in the document. See HistoryState for more information on `HistoryState` objects.

Note: Because the `HistoryStates` class is also a property of the Document object, you use the property name, `historyStates`, rather than the class name, `HistoryStates`, in your code.

The following example uses correct syntax to fill a Selection object (referred to by the variable selRef) with an object in the `HistoryStates` collection:

```
selRef.fill(activeDocument.historyStates[7])
```

The following example, which uses an upper case *H* in the object name, is incorrect:

```
selRef.fill(activeDocument.HistoryStates[7])
```

Properties

Property	Value Type	What it is
`length`	`number (long)`	Read-only. The number of elements in the `HistoryStates` collection.
`parent`	`object (Document)`	Read-only. The `HistoryStates` object's container.
`typename`	`string`	Read-only. The class name of the referenced `HistoryStates` object.

Methods

Method	Parameter Type	Returns	What it does
`index` `(itemKey)`	`number`	`HistoryState`	Gets an element from the `HistoryStates` collection.
`getByName` `(name)`	`string`	`HistoryState`	Get the first element in the `HistoryStates` collection with the provided name.

HSBColor

Options that can be specified for a color object using the HSB color model.

Properties

Property	Value Type	What it is
`brightness`	`number (double)`	Read-write. The brightness value (between 0.0 and 100.0).
`hue`	`number (double)`	Read-write. The hue value (between 0.0 and 360.0).
`saturation`	`number (double)`	Read-write. The saturation value (between 0.0 and 100.0).
`typename`	`string`	Read-only. The class name of the referenced `HSBColor` object.

IndexedConversionOptions

Options that can be specified when converting an RGB image to an indexed color model.

Properties

Property	Value Type	What it is

colors	number (long)	Read-write. The number of palette colors.
		Note: Valid only when
		palette = Palette.LOCALADAPTIVE;
		palette = Palette.LOCALPERCEPTUAL;
		palette = Palette.LOCALSELECTIVE;
		palette = Palette.MACOSPALETTE;
		palette = Palette.UNIFORM;
		palette = Palette.WEBPALETTE; or
		palette = Palette.WINDOWSPALETTE .
		See palette.

dither	Dither	Read-write. The dither type.
ditherAmount	number (long)	Read-write. The amount of dither. (1 - 100). **Note:** Valid only when dither = Dither.diffusion.
forced	ForcedColors	Read-write. The type of colors to force into the color palette.
matte	MatteType	Read-write. The color to use to fill anti-aliased edges adjacent to transparent areas of the image (default: MatteType.WHITE). **Note:** When transparency = false, the matte color is applied to transparent areas. See transparency.
palette	Palette	Read-write. The palette type (default: Palette.EXACT).
preserveExactColors	boolean	Read-write. Indicates whether to protect colors in the image that contain entries in the color table from being dithered. **Note:** Valid only when dither = Dither.DIFFUSION. See dither.
transparency	boolean	Read-write. Indicates whether to preserve transparent areas of the image during conversion to GIF format.
typename	string	Read-only. The class name of the referenced IndexedConversionOptions object.

JPEGSaveOptions

Options that can be specified when saving a document in JPEG format.

Properties

Property	Value Type	What it is
`embedColorProfile`	`boolean`	Read-write. Indicates whether to embed the color profile in the document.
`formatOptions`	`FormatOptions`	Read-write.The download format to use (default: `FormatOptions.STANDARDBASELINE`).
`matte`	`MatteType`	Read-write. The color to use to fill anti-aliased edges adjacent to transparent areas of the image (default: `MatteType.WHITE`). **Note:** When `transparency = false`, the matte color is applied to transparent areas. See transparency.
`quality`	`number (long)`	Read-write. The image quality setting to use (affects file size and compression) (0 - 12; default: `3`).
`scans`	`number (long)`	Read-write. The number of scans to make to incrementally display the image on the page (3 - 5; default: `3`). **Note:** Valid only for when `formatOptions = FormatOptions.PROGRESSIVE`.
`typename`	`string`	Read-only. The class name of the referenced `JPEGSaveOptions` object.

LabColor

Options that can be specified when defining a color object using the LAB color model.

Properties

Property	Value Type	What it is
`a`	`number (double)`	Read-write. The a-value (-128.0 - 127.0).
`b`	`number (double)`	Read-write. The b-value (-128.0 - 127.0).
`l`	`number (double)`	Read-write. The L-value (0.0 - 100.0).
`typename`	`string`	Read-only. The class name of the referenced `LabColor` object.

LayerComp

A snapshot of a state of the layers in a document (can be used to view different page layouts or compostions).

Note: Because the `LayerComp` class is also a property of the Document object, you use the property name, `layerComp`, rather than the class name, `LayerComp`, in your code.

The following example uses correct syntax to set the comment property value for a `LayerComp` object named `myLayerComp`:

```
activeDocument.layerComp("myLayerComp").comment = "View from shoreline"
```

The following example, which uses an upper case *L* in the object name, is incorrect:

```
activeDocument.LayerComp("myLayerComp").comment = "View from shoreline"
```

Properties

Property	Value Type	What it is
appearance	boolean	Read-write. Indicates whether to use layer appearance (layer styles) settings.
comment	string	Read-write. A description of the layer comp.
name	string	Read-write. The name of the layer comp.
parent	object (Document)	Read-write. The `layerComp` object's container.
position	boolean	Read-write. Indicates whether to use layer position.
selected	boolean	Read-only. Indicates whether the layer comp is currently selected.
typename	string	Read-only. The class name of the referenced `layerComp` object.
visibility	boolean	Read-write. Indicates whether to use layer visibility settings .

Methods

Method	Parameter Type	Returns	What it does
apply ()			Applies the layer comp to the document.
recapture ()			Recaptures the current layer state(s) for this layer comp.

Method	Parameter Type	Returns	What it does (Continued)
remove ()			Deletes the `layerComp` object.
resetfromComp ()			Resets the layer comp state to the document state.

LayerComps

The collection of `layerComp` objects in the document. See LayerComp for information on `layerComp` objects.

Note: Because the `LayerComps` class is also a property of the Document object, you use the property name, `layerComps`, rather than the class name, `LayerComps`, in your code.

The following example uses correct syntax to add a `LayerComps`:

```
activeDocument.layerComps.add("myLayerComp", "View from Shoreline", true, true,
true)
```

The following example, which uses an upper case *L* in the object name, is incorrect:

```
activeDocument.LayerComps.add("myLayerComp", "View from Shoreline", true, true,
true)
```

Properties

Property	Value Type	What it is
length	number (long)	Read-only. The number of elements in the `layerComps` collection.
parent	object (Document)	Read-only. The `layerComps` object's container.
typename	string	Read-only. The class name of the referenced `layerComps` object.

Methods

Method	Parameter Type	Returns	What it does
index (itemKey)	number	LayerComp	Gets an element from the `layerComps` collection.
add (name, comment, appearance, position, visibility)	string string boolean boolean boolean	LayerComp	Adds a layer comp.
getByName (name)	string	LayerComp	Gets the first element in the collection with the provided name.
removeAll ()			Removes all `layerComp` objects from the `layerComps` collection.

Layers

The collection of layer objects, including ArtLayer and LayerSet objects, in the document.

Note: Because the `Layers` object is a property of the Document object (as well as several other objects), you use the property name, `layers`, rather than the class name, `Layers`, in your code. The following example uses the `length` property to count the number of `layer` objects in the active document, then displays the number on the screen:

```
var layerNum = app.activeDocument.layers.length
alert(layerNum)
```

The following example uses an upper case *L*, which is incorrect:

```
var layerNum = app.activeDocument.Layers.length
alert(layerNum)
```

Properties

Property	Value Type	What it is
length	number (long)	Read-only. The number of elements in the `layers` collection.
parent	object (document or layerSet)	Read-only. The `layers` object's container.
typename	string	Read-only. The class name of the referenced `layers` object.

Methods

Method	Parameter Type	Returns	What it does
index (itemKey)	number	object(Layer)	Gets an element from the collection.
getByName (name)	string	Layer	Gets the first element in the `layers` collection with the provided name.
removeAll ()			Removes all layers from the collection.

LayerSet

A group of layer objects, which can include `artLayer` objects and other (nested) `layerSet` objects. A single command or set of commands manipulates all layers in a `layerSet` object.

Note: Most likely, you will use variables to refer to `layerSet` objects in your script. However, if you choose not to use a variable, be aware that, because the `LayerSet` class is also a property of the Document object, you use the property name, `layerSet`, rather than the class name, `LayerSet`, in your code.

The following example uses correct syntax to refer to a `layerSet` object by name and then assign its `allLocked` property value:

```
documents(0).layerSet("myLayerSet").allLocked = true
```

The following example, which uses an upper case *L* in the object name, is incorrect:

```
documents(0).LayerSet("myLayerSet").allLocked = true
```

Properties

Property	Value Type	What it is
allLocked	boolean	Read-write. Indicates whether the contents in the layers contained in the `layerSet` object are editable.
artLayers	ArtLayers	Read-only. The `artLayer` objects in this layer set.
blendMode	BlendMode	Read-write. The blend mode to use for the layer set.
bounds	array(UnitValue)	Read-only. The bounding rectangle of the layer set. **Note:** For information about the `UnitValue` type, see the *JavaScript Tools Guide*.
enabledChannels	array of Channel objects	Read-write. The channels enabled for the layer set; must be a list of component channels. **Note:** See `kind` in the Properties table for the Channel object (Channel).
layers	Layers	Read-only. The layers in this `layerSet` object.
layerSets	LayerSets	Read-only. Layer Sets contained within a Layer Set.
linkedLayers	array of ArtLayer and/or LayerSet	Read-only. The layers linked to this `layerSet` object.
name	string	Read-write. The name of the `layerSet` object.
opacity	number (double)	Read-write. The master opacity of the `layerSet` object (0.0 - 100.0).
parent	object (Document or LayerSet)	Read-only. The `layerSet` object's container.
typename	string	Read-only. The class name of the referenced `layerSet` object.
visible	boolean	Read-write. Indicates whether the `layerSet` object is visible.

Methods

Method	Parameter Type	Returns	What it does
duplicate ([relativeObject] [, insertionLocation])	object (ArtLayer or LayerSet) ElementPlacement	LayerSet	Creates a duplicate of the `layerSet` object.
link (with)	object (ArtLayer or LayerSet)		Links the layer set with another layer.
merge ()		ArtLayer	Merges the layerset; returns a reference to the art layer created by this method.
move (relativeObject, insertionLocation)	object (ArtLayer or LayerSet) ElementPlacement		Moves the `layerSet` object.
remove ()			Deletes the `layerSet` object.
resize ([horizontal] [, vertical] [, anchor])	number (double) number (double) AnchorPosition		Resizes all layers in the layer set to to the specified dimensions (as a percentage of its current size) and places the layer set in the specified position.
rotate (angle [, anchor])	number (double) AnchorPosition		Rotates all layers in the layer set around the specified anchor point (default: `AnchorPosition.MIDDLECENTER`).
translate ([deltaX] [, deltaY])	UnitValue UnitValue		Moves the position relative to its current position. **Note:** For more information about the UnitValue type, see the *JavaScript Tools Guide*
unlink ()			Unlinks the layer set.

LayerSets

The collection of `layerSet` objects in the document. See LayerSet for information on `layerSet` objects.

Note: Because the `LayerSets` class is a property of the Document object, you use the property name, `layerSets`, rather than the class name, `LayerSets`, in your code. For example:

```
var laysetRef = docRef.layerSets.add()
```

The following sample uses the `layerSets` object incorrectly:

```
var laysetRef = docRef.LayerSets.add()
```

Properties

Property	Value Type	What it is
`length`	`number (long)`	Read-only. The number of elements in the `LayerSets` collection.
`parent`	`object (Document or LayerSet)`	Read-only. The `layerSets` object's container.
`typename`	`string`	Read-only. The class name of the referenced `layerSets` object.

Methods

Method	Parameter Type	Returns	What it does

`index`
`(itemKey)`
	`number`	`LayerSet`	Gets an element from the `layerSets` collection.

`add`
`()`
		`LayerSet`	Creates a new `layerSet` object.

`getByName`
`(name)`
	`string`	`LayerSet`	Gets the first element in the `layerSets` collection with the provided name.

`removeAll`
`()`
			Removes the layer set, and any layers or layer sets it contains, from the document.

Sample Script

The following script creates three layer sets, then nests a second layer set in each layer set, and then creates a text layer in each nested set that that displays the text “Layer in *n* Set Inside *n* Set”, where *n* represents the ordinal number of the set (first, second, or third).

Note: The script uses the ExtendScript `$` object. For further details, see the *JavaScript Tools Guide*.

LayerSets.jsx

```
$.level = 1

//close all open documents
while (app.documents.length) {
   app.activeDocument.close()
}
```

```
// create a working document
var docRef = app.documents.add()

// create an array to hold the layer sets
var myLayerSets = new Array()

// Create an array to hold the text
var textArray = Array("First", "Second", "Third")

//Create an indexer variable
var i = 0

// Create three layer sets at the top level
for (i = 0; i < 3; i++) {
   myLayerSets[i] = new Array()
   myLayerSets[i][0] = docRef.layerSets.add()
}

// Rearrange the layer sets with the first one on top, second next, etc.
myLayerSets[1][0].moveAfter(myLayerSets[0][0])
myLayerSets[2][0].moveAfter(myLayerSets[1][0])

// Create a layer set inside each layer set
for (i = 0; i < 3; i++) {
   myLayerSets[i][0].name = textArray[i] + " Set"
   myLayerSets[i][1] = myLayerSets[i][0].layerSets.add()
   myLayerSets[i][1].name = "Inside " + textArray[i] + " Set"
}

// Create an array to hold the layers
var myLayers = new Array()

// Create a text layer with a description inside each layer set
for (i = 0; i < 3; i++) {
   myLayers[i] = myLayerSets[i][1].artLayers.add()
   myLayers[i].kind = LayerKind.TEXT
   myLayers[i].textItem.contents = "Layer in " + textArray[i] + " Set Inside "
        + textArray[i] + " Set"
   myLayers[i].textItem.position = Array(app.activeDocument.width * i * 0.33,
          app.activeDocument.height * (i + 1) * 0.25)
   myLayers[i].textItem.size = 12
}
```

MeasurementLog

The measurement log for the application. See measurementLog (in the Properties table for the Application object.)

Note: The MeasurementLog feature is available in the Extended Version only.

Because the `MeasurementLog` class is a property of the `Application` object, you use the property name, `measurementLog`, rather than the class name, `MeasurementLog`, in your code.

Methods

Method	Parameter type	Returns	What it does

`exportMeasurements`
`([file]`
`[, range])`
`[, dataPoints])`
	`File`
	`MeasurementRange`
	`array of strings`
			Export some measurement(s).

`deleteMeasurements`
`([range])`
	`MeasurementRange`		Delete a measurement.

MeasurementScale

The measurement scale for the document. See measurementScale (in the Properties table for the Document object.)

Note: The MeasurementScale feature is available in the Extended Version only.

Because the `MeasurementScale` class is a property of the `Document` object, you use the property name, `measurementScale`, rather than the class name, `MeasurementScale`, in your code. For example:

```
activeDocument.measurementScale.pixelLength = 25
```

The following code incorrectly uses an upper case *M*:

```
activeDocument.MeasurementScale.pixelLength = 25
```

Properties

Property	Value Type	What it is
`pixelLength`	`number (long)`	Read-write. The length in pixels this scale equates to.
`logicalLength`	`number (double)`	Read-write. The logical length this scale equates to.
`logicalUnits`	`string`	Read-write. The logical units for this scale.

NoColor

An object that represents a missing color.

Properties

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

Notifier

An event-handler object that tells the script to execute specified code when a specified event occurs.

Note: For notifiers to work, they must be enabled. See the notifiersEnabled property of the Application object.

Note: Events that occur within scripts do not generally trigger notifiers, because they occur inside of a "play script" event.

Note: Because the `Notifier` class is also a property of the Application object, you use the property name, `notifier`, rather than the class name, `Notifier`, in your code.

Properties

Property	Value type	What it is
`event`	`string`	Read-only. The event ID in four characters or a unique string that the notifier is associated with. **Note:** For a list of four-character codes, see Appendix A: Event ID Codes.
`eventClass`	`string`	Read-only. The class ID associated with the `event` for the `Notifier` object, four characters or a unique string. **Note:** When an event applies to multiple types of objects, you use this propery to distinguish which object this `Notifier` applies to. For example, the Make event (`"Mk "`)applies to documents (`"Dcmn"`), channels (`"Chnl"`) and other objects.
`eventFile`	`File`	Read-only. The path to the file to execute when the event occurs/activates the notifier. **Note:** For information about the `File` object, see the *JavaScript Tools Guide*
`parent`	`object (Application)`	Read-only. The `notifier` object's container.
`typename`	`string`	Read-only. The class name of the referenced `notifier` object.

Methods

Method	Parameter type	Returns	What it does

`remove`
`()`
			Deletes the `notifier` object.
			Note: You can remove a `notifier` object from the Script Events Manager drop-down list by deleting the file named `Script Events Manager.xml` from in the Photoshop preferences folder. See Adobe Photoshop CS3 help for more information.

Notifiers

The collection of `notifier` objects in the document; the `notifiers` property of the `app` object. See 'Notifier' on page 135 for information on `notifier` objects. See notifiers (in the Properties table of the `app` object).

In order to enable notifiers to run scripts

Note: Because the `Notifiers` class is a property of the Application object, you use the property name, `notifiers`, rather than the class name, `Notifiers`, in your code. For example:

```
var notRef = app.notifiers.add("OnClickGoButton", eventFile)
```

The following sample uses the `Notifiers` object incorrectly:

```
var notRef = app.Notifiers.add("OnClickGoButton", eventFile)
```

Properties

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

Methods

Method	Parameter type	Returns	What it does
index (itemKey)	number	Notifier	Gets an element from the `notifiers` collection.
add (event, eventFile [, eventClass])	string File string	Notifier	Creates a `notifier` object. `event` defines the class ID of the event: four characters or a unique string. For a list of four-character codes, see Appendix A: Event ID Codes. **Tip:** Remember to omit the single quotes when including a four-character ID in your code. `eventFile` defines the script file that executes when the event occurs. **Note:** An `eventClass` value corresponds to the class of object the event is applied to: four characters or a unique string. When an `event` applies to multiple types of objects, you use the `eventClass` parameter to distinguish which object this `Notifier` applies to. For example, the Make event (`"Mk  "`) applies to documents (`"Dcmn"`), channels (`"Chnl"`) and other objects.
removeAll ()			Removes all `notifier` objects from the `notifiers` collection. **Note:** You can remove a `notifier` object from the Script Events Manager drop-down list by deleting the file named `Script Events Manager.xml` from in the Photoshop preferences folder. See Adobe Photoshop CS3 help for more information.

PathItem

A path or drawing object, such as the outline of a shape or a straight or curved line, which contains sub paths that comprise its geometry.

Note: Because the `PathItem` class is also a property of the Document object, you use the property name, `pathItem`, rather than the class name, `PathItem`, in your code.

The following example uses correct syntax to select a `pathItem` object :

```
activeDocument.pathItem(“myPath”).select()
```

The following example, which uses an upper case *P* in the object name, is incorrect:

```
activeDocument.PathItem(“myPath”).select()
```

Properties

Property	Value Type	What it is
kind	PathKind	Read-write. The `pathItem` object’s type.
name	string	Read-write. The `pathItem` object’s name.
parent	object (Document)	Read-only. The `pathItem` object's container.
SubPathItems	SubPathItems	Read-only. The sub path objects for this `pathItem` object.
typename	string	Read-only. The class name of the referenced `pathItem` object.

Methods

Method	Parameter Type	Returns	What it does
deselect ()			Deselects this `pathItem` object.
duplicate (name)	string		Duplicates this `pathItem` object with the new name specified in the argument.
fillPath ([fillColor] [, mode] [, opacity] [, preserveTransparency] [, feather] [, wholePath] [, antiAlias])	Object (SolidColor, ArtLayer, HistoryState) ColorBlendMode number (double) boolean number (double) boolean boolean		Fills the area enclosed by the path (`opacity`: 0 - 100 as percent; `feather`: 0.0 - 250.0 in pixels). The `wholePath` parameter indicates that all subpaths are used when doing the fill. (Default:`true`).

Method	Parameter Type	Returns	What it does (Continued)
makeClippingPath ([flatness])	number (double)		Makes this pathItem object the clipping path for this document; the optional parameter tells the PostScript printer how to approximate curves in the path (0.2 - 100).
makeSelection ([feather] [, antiAlias] [, operation])	number (double) boolean SelectionType		Makes a selection object, whose border is the path, from this pathItem object (feather: 0.0 - 250.0 in pixels). **Note:** See Selection.
remove ()			Deletes this pathItem object.
select ()			Makes this pathItem object the active or selected pathItem object.
strokePath ([tool] [, simulatePressure])	ToolType boolean		Strokes the path with the specified information.

Sample Script

The following creates a path in three segments: two diagonal lines that form a *V*, and a curved line above the *V* that makes it look like a 2D ice cream cone.

Paths.jsx

```
// Save the current preferences
var startRulerUnits = app.preferences.rulerUnits
var startTypeUnits = app.preferences.typeUnits
var startDisplayDialogs = app.displayDialogs

// Set Adobe Photoshop CS3 to use pixels and display no dialogs
app.preferences.rulerUnits = Units.PIXELS
app.preferences.typeUnits = TypeUnits.PIXELS
app.displayDialogs = DialogModes.NO

// first close all the open documents
while (app.documents.length) {
   app.activeDocument.close()
}

// create a document to work with
var docRef = app.documents.add(5000, 7000, 72, "Simple Line")

//line 1--it’s a straight line so the coordinates for anchor, left, and right
//for each point have the same coordinates
var lineArray = new Array()
   lineArray[0] = new PathPointInfo
   lineArray[0].kind = PointKind.CORNERPOINT
   lineArray[0].anchor = Array(100, 100)
   lineArray[0].leftDirection = lineArray[0].anchor
   lineArray[0].rightDirection = lineArray[0].anchor

   lineArray[1] = new PathPointInfo
   lineArray[1].kind = PointKind.CORNERPOINT
   lineArray[1].anchor = Array(150, 200)
   lineArray[1].leftDirection = lineArray[1].anchor
   lineArray[1].rightDirection = lineArray[1].anchor

var lineSubPathArray = new Array()
   lineSubPathArray[0] = new SubPathInfo()
   lineSubPathArray[0].operation = ShapeOperation.SHAPEXOR
   lineSubPathArray[0].closed = false
   lineSubPathArray[0].entireSubPath = lineArray

// line 2
var lineArray2 = new Array()
   lineArray2[0] = new PathPointInfo
   lineArray2[0].kind = PointKind.CORNERPOINT
   lineArray2[0].anchor = Array(150, 200)
   lineArray2[0].leftDirection = lineArray2[0].anchor
   lineArray2[0].rightDirection = lineArray2[0].anchor

   lineArray2[1] = new PathPointInfo
   lineArray2[1].kind = PointKind.CORNERPOINT
   lineArray2[1].anchor = Array(200, 100)
   lineArray2[1].leftDirection = lineArray2[1].anchor
   lineArray2[1].rightDirection = lineArray2[1].anchor

   lineSubPathArray[1] = new SubPathInfo()
   lineSubPathArray[1].operation = ShapeOperation.SHAPEXOR
   lineSubPathArray[1].closed = false
   lineSubPathArray[1].entireSubPath = lineArray2

//ice cream curve
//it’s a curved line, so there are 3 points, not 2
//coordinates for the middle point (lineArray3[1]) are different.
//The left direction is positioned “above” the anchor on the screen.
//The right direction is positioned “below” the anchor
//You can change the coordinates for these points to see
//how the curve works...
var lineArray3 = new Array()
   lineArray3[0] = new PathPointInfo
   lineArray3[0].kind = PointKind.CORNERPOINT
   lineArray3[0].anchor = Array(200, 100)
   lineArray3[0].leftDirection = lineArray3[0].anchor
   lineArray3[0].rightDirection = lineArray3[0].anchor

   lineArray3[1] = new PathPointInfo
   lineArray3[1].kind = PointKind.CORNERPOINT
   lineArray3[1].anchor = Array(150, 50)
   lineArray3[1].leftDirection = Array(100, 50)
   lineArray3[1].rightDirection = Array(200, 50)

   lineArray3[2] = new PathPointInfo
   lineArray3[2].kind = PointKind.CORNERPOINT
   lineArray3[2].anchor = Array(100, 100)
   lineArray3[2].leftDirection = lineArray3[2].anchor
   lineArray3[2].rightDirection = lineArray3[2].anchor
   lineSubPathArray[2] = new SubPathInfo()
   lineSubPathArray[2].operation = ShapeOperation.SHAPEXOR
   lineSubPathArray[2].closed = false
   lineSubPathArray[2].entireSubPath = lineArray3

//create the path item
var myPathItem = docRef.pathItems.add("A Line", lineSubPathArray)

// stroke it so we can see something
myPathItem.strokePath(ToolType.BRUSH)

// Reset the application preferences
preferences.rulerUnits = startRulerUnits
preferences.typeUnits = startTypeUnits
displayDialogs = startDisplayDialogs
```

PathItems

The collection of pathItem objects in the document. See PathItem for information on pathItem objects.

Note: Because the PathItems class is a property of the Document object, you use the property name, pathItems, rather than the class name, PathItems, in your code. For example:

```
var myPathItem = docRef.pathItems.add("A Line", lineSubPathArray)
```

The following sample uses the PathItems object incorrectly:

```
var myPathItem = docRef.PathItems.add("A Line", lineSubPathArray)
```

Properties

Property	Value Type	What it is
length	number (long)	Read-only. The number of pathItem objects in the pathItems collection.
parent	object (document)	Read-only. The pathItems object's container.
typename	string	Read-only. The class name of the referenced pathItems object.

Methods

Method	Parameter Type	Returns	What it does
index (itemKey)	number	PathItem	Gets a pathItem object from the pathItems collection.
add (name, entirePath)	string array of SubPathInfo objects	PathItem	Creates a new PathItem object from the sub paths defined in the array provided in the entirePath parameter. A new SubPathItem object is created for each SubPathInfo object provided in entirePath, and those SubPathItem objects are added to the SubPathItems collection of the returned PathItem.
getByName (name)	string	PathItem	Get the first element in the pathItems collection with the provided name.
removeAll ()			Removes all pathItem objects from the pathItems collection.

PathPoint

Information about an array of `PathPointInfo` objects.

Note: You do not use the `PathPoint` object to create points that make up a path. Rather, you use the `PathPoint` object to retrieve information about the points that describe path segments. To create path points, use the `PathPointInfo` objects. See PathPointInfo.

Properties

Property	Value Type	What it is
anchor	array(UnitValue)	Read-only. The point on the curve (leftDirection/rightDirection are points representing the control handle end points). **Note:** For information about the `UnitValue` type, see the *JavaScript Tools Guide*.
kind	PointKind	Read-only. The `PathPoint` object's type.
leftDirection	array(UnitValue)	Read-only. The x and y coordinates that define the left handle. **Note:** For information about the `UnitValue` type, see the *JavaScript Tools Guide*.
parent	object (SubPathItem)	Read-only. The `PathPoint` object's container.
rightDirection	array(UnitValue)	Read-only. The x and y coordinates that define the right handle. **Note:** For information about the `UnitValue` type, see the *JavaScript Tools Guide*.
typename	string	Read-only. The class name of the referenced `PathPoint` object.

PathPointInfo

A point on a path, expressed as an array of three coordinate arrays: the anchor point, left direction point, and right direction point. For paths that are straight segments (not curved), the coordinates of all three points are the same. For curved segements, the the coordinates are different. The difference between the anchor point and the left or right direction points determines the arc of the curve. You use the left direction point to bend the curve “outward” or make it convex; you use the right direction point to bend the curve “inward” or make it concave.

Properties

Property	Value Type	What it is
anchor	array	Read-write. The x and y coordinates of one end point of the path segment.
kind	PointKind	Read-write. The PathPointInfo object’s kind.
leftDirection	array of UnitValue	Read-write. The location of the left direction point (’in’ position). **Note:** For information about the UnitValue type, see the *JavaScript Tools Guide*.
rightDirection	array of UnitValue	Read-write. The location of the right handle (’out’ position). **Note:** For information about the UnitValue type, see the *JavaScript Tools Guide*.
typename	string	Read-only. The class name of the referenced PathPointInfo object.

PathPoints

A collection of PathPoint objects that comprises the PathPoints property of the SubPathItem object. See SubPathItem for more information.

Properties

Property	Value Type	What it is
length	number (long)	Read-only. The number of elements in the PathPoints collection.
parent	object (SubPathItem)	Read-only. The PathPoints object's container.
typename	string	Read-only. The class name of the referenced PathPoints object.

Methods

Method	Parameter type	Returns	What it does
index (itemKey)	number	PathPoint	Gets an element from the PathPoints collection.

PDFOpenOptions

Options that can be specified when opening a document in generic Adobe PDF format.

Properties

Property	Value Type	What it is
antiAlias	boolean	Read-write. Indicates whether to use antialias.
bitsPerChannel	BitsPerChannelType	Read-write. The number of bits per channel.
constrainProportions	boolean	Deprecated for Adobe Photoshop CS3.
cropPage	CropToType	Read-write. The method of cropping to use.
height	UnitValue	Deprecated for Adobe Photoshop CS3.
mode	OpenDocumentMode	Read-write. The color model to use.
name	string	Read-write. The name of the document.
page	number (long)	Read-write. The page to which to open the document.
resolution	number (double)	Read-write. The resolution of the document (in pixels per inch).
suppressWarnings	boolean	Read-write. Indicates whether to suppress warnings when opening the document.
typename	string	Read-only. The class name of the referenced PDFOpenOptions object.
usePageNumber	boolean	Read-write. Indicates whether the value specified in the page property will refer to an image number when usePageNumber = false. See page.
width	UnitValue	Deprecated for Adobe Photoshop CS3.

PDFSaveOptions

Options that can be specified when saving a document in Adobe PDF format.

Properties

Property	Value Type	What it is
alphaChannels	boolean	Read-write. Indicates whether to save the alpha channels with the file.
annotations	boolean	Read-write. Indicates whether to save comments with the file.
colorConversion	boolean	Read-write. Indicates whether to convert the color profile to a destination profile.
convertToEightBit	boolean	Read-write. Indicates whether to convert a 16-bit image to 8-bit for better compatibility with other applications.
description	string	Read-write. Description of the save options to use.
destinationProfile	string	Read-write. Description of the final RGB or CMYK output device, such as a monitor or a press standard.
downgradeColorProfile	boolean	Deprecated for Adobe Photoshop CS3.
downSample	PDFResample	Read-write. The down sample method to use.
downSampleSize	number (double)	Read-write. The size to downsample images if they exceed the limit in pixels per inch.
downSampleSizeLimit	number (double)	Read-write. Limits downsampling or subsampling to images that exceed this value in pixels per inch.
embedColorProfile	boolean	Read-write. Indicates whether to embed the color profile in the document.
embedFonts	boolean	Deprecated for Adobe Photoshop CS3.
embedThumbnail	boolean	Read-write. Indicates whether to include a small preview image in Adobe PDF files.
encoding	PDFEncoding	Read-write. The encoding method to use (default: PDFEncoding.PDFZIP).
interpolation	boolean	Deprecated for Adobe Photoshop CS3.

Property	Value Type	What it is (Continued)
jpegQuality	number (long)	Read-write. The quality of the produced image (0 - 12), which is inversely proportionate to the compression amount. **Note:** Valid only when encoding = PDFEncoding.JPEG.
layers	boolean	Read-write. Indicates whether to save the document's layers.
optimizeForWeb	boolean	Read-write. Indicates whether to improve performance of PDF files on Web servers.
outputCondition	string	Read-write. An optional comment field for inserting descriptions of the output condition. The text is stored in the PDF/X file.
outputConditionID	string	Read-write. Indentifier for the output condition.
PDFCompatibility	PDFCompatibility	Read-write. The PDF version to make the document compatible with.
PDFStandard	PDFStandard	Read-write. The PDF standard to make the document compatible with.
preserveEditing	boolean	Read-write. Indicates whether to reopen the PDF in Adobe Photoshop CS3 with native Photoshop data intact.
presetFile	string	Read-write. The preset file to use for settings. **Note:** This option overrides other settings.
profileInclusionPolicy	boolean	Read-write. Indicates whether to show which profiles to include.
registryName	string	Read-write. URL where the output condition is registered.
spotColors	boolean	Read-write. Indicates whether to save spot colors.
tileSize	nunber (long)	Read-write. Compression option. **Note:** Valid only when encoding = PDFEncoding.JPEG2000.
transparency	boolean	Deprecated for Adobe Photoshop CS3.
typename	string	Read-only. The class name of the referenced PDFSaveOptions object.
useOutlines	boolean	Deprecated for Adobe Photoshop CS3.

Property	Value Type	What it is (Continued)
vectorData	boolean	Deprecated for Adobe Photoshop CS3.
view	boolean	Read-write. Indicates whether to open the saved PDF in Adobe Acrobat.

PhotoCDOpenOptions

Deprecated in Adobe Photoshop CS3. Kodak PhotoCD is now found in the Goodies folder on the Adobe Photoshop CS3 Install DVD.

Options to be specified when opening a Kodak Photo CD (PCD) files, including high-resolution files from Pro Photo CD discs.

Properties

Property	Value Type	What it is
colorProfileName	string	Read-write. The profile to use when reading the image.
colorSpace	PhotoCDColorSpace	Read-write. The colorspace for the image.
orientation	Orientation	Read-write. The image orientation.
pixelSize	PhotoCDSize	Read-write. The image dimensions.
resolution	number (double)	Read-write. The image resolution (in pixels per inch).
typename	string	Read-only. The class name of the referenced photoCDOpenOptions object.

PhotoshopSaveOptions

Options that can be specified when saving a document in PSD format.

Properties

Property	Value Type	What it is
`alphaChannels`	`boolean`	Read-write. Indicates whether to save the alpha channels.
`annotations`	`boolean`	Read-write. Indicates whether to save the annotations.
`embedColorProfile`	`boolean`	Read-write. Indicates whether to embed the color profile in the document.
`layers`	`boolean`	Read-write. Indicates whether to preserve the layers.
`spotColors`	`boolean`	Read-write. Indicates whether to save the spot colors.
`typename`	`string`	Read-only. The class name of the referenced `photoshopSaveOptions` object.

PICTFileSaveOptions

Options that can be specified when saving a document in PICT format.

Properties

Property	Value Type	What it is
alphaChannels	boolean	Read-write. Indicates whether to save the alpha channels.
compression	PICTCompression	Read-write. (default: PICTCompression.NONE)
embedColorProfile	boolean	Read-write. Indicates whether to embed the color profile in the document.
resolution	PICTBitsPerPixels	Read-write. The number of bits per pixel.
typename	string	Read-only. The class name of the referenced PICTFileSaveOptions object.

PICTResourceSaveOptions

Options that can be specified when saving a document as a PICT Resource file.

Properties

Property	Value Type	What it is
alphaChannels	boolean	Read-write. Indicates whether to save the alpha channels.
compression	PICTCompression	Read-write. The type of compression to use (default: PICTCompression.NONE).
embedColorProfile	boolean	Read-write. Indicates whether to embed the color profile in the document.
name	string	Read-write. The name of the PICT resource.
resolution	PICTBitsPerPixels	Read-write. The number of bits per pixel.
resourceID	number (long)	Read-write. The ID of the PICT resource (default: 128).
typename	string	Read-only. The class name of the referenced PICTResourceSaveOptions object.

PicturePackageOptions

Options that can be specified for a Picture Package.

Properties

Property	Value type	What it is
content	PicturePackageTextType	Read-write. The content information (default: PicturePackageTextType.NONE).
flatten	boolean	Read-write. Indicates whether all layers in the final document are flattened (default: true).
font	GalleryFontType	Read-write. The font used for security text (default: GalleryFontType.ARIAL).
fontSize	number (long)	Read-write. The font size used for security text (default: 12).
layout	string	Read-write. The layout to use to generate the picture package (default: "(2)5x7").
mode	NewDocumentMode	Read-write. Read-write. The color profile to use as the document mode (default: NewDocumentMode.RGB).
opacity	number (long)	Read-write. The web page security opacity as a percent (default: 100).
resolution	number (double)	Read-write. The resolution of the document in pixels per inch (default: 72.0).
text	string	Read-write. The picture package custom text. **Note:** Valid only when content = PicturePackageType.USER. See content.
textColor	RGBColor	Read-write. The color to use for security text.
textPosition	GallerySecurityTextPositionType	Read-write. The security text position (default: GallerySecurityTextPositionType.CENTERED).
textRotate	GallerySecurityTextRotateType	Read-write. The orientation to use for security text (default: GallerySecurityTextRotateType.ZERO).
typename	string	Read-only. The class name of the referenced PicturePackageOptions object.

PixarSaveOptions

Options that can be specified when saving a document in Pixar format.

Properties

Property	Value Type	What it is
`alphaChannels`	`boolean`	Read-write. Indicates whether to save the alpha channels.
`typename`	`string`	Read-only. The class name of the referenced `PixarSaveOptions` object.

PNGSaveOptions

Options that can be specified when saving a document in PNG format.

Properties

Property	Value Type	What it is
`interlaced`	`boolean`	Read-write. Indicates whether the should rows be interlaced (default: `false`).
`typename`	`string`	Read-only. The class name of the referenced `PNGSaveOptions` object.

Preferences

Options to define for the `preferences` property of the `app` object. See preferencesFolder (in the Properties table for the `app` object).

Note: Because the `Preferences` class is a property of the Application object, you use the property name, `preferences`, rather than the class name, `Preferences`, in your code. For example:

```
app.preferences.rulerUnits = Units.PIXELS
app.preferences.typeUnits = TypeUnits.PIXELS
```

The following code incorrectly uses an upper case *P*:

```
app.Preferences.rulerUnits = Units.PIXELS
app.Preferences.typeUnits = TypeUnits.PIXELS
```

Note: Defining the `preferences` properties is basically equivalent to selecting Edit > Preferences (Windows) or Photoshop > Preferences in the Adobe Photoshop CS3 application. For explanations of individual settings, please refer to Adobe Photoshop CS3 Help.

Properties

Property	Value Type	What it is
additionalPluginFolder	File	Read-write. The path to an additional plug-in folder. **Note:** Valid only when `useAdditionalPluginFolder = true`. See useAdditionalPluginFolder. **Note:** For information about the File object, see the *JavaScript Tools Guide*
appendExtension	SaveBehavior	Read-write. Save files with extensions on Windows.
askBeforeSavingLayeredTIFF	boolean	Read-write. Indicates whether to ask the user to verify layer preservation options when saving a file in TIFF format.
autoUpdateOpenDocuments	boolean	Read-write. Indicates whether to automatically update open documents.
beepWhenDone	boolean	Read-write. Indicates whether to beep when a process finishes.
colorChannelsInColor	boolean	Read-write. Indicates whether to display component channels in the Channels palette in color.
colorPicker	ColorPicker	Read-write.
columnGutter	number (double)	Read-write. The width of the column gutters (in points). (0.1 - 600.0).

Property	**Value Type**	**What it is (Continued)**
`columnWidth`	`number (double)`	Read-write. Column width (in points) (0.1 - 600.0).
`createFirstSnapshot`	`boolean`	Read-write. Indicates whether to automatically make the first snapshot when a new document is created.
`dynamicColorSliders`	`boolean`	Read-write. Indicates whether dynamic color sliders appear in the Color palette.
`editLogItems`	`EditLogItemsType`	Read-write. The options for editing history log items. **Note:** Valid only when `useHistoryLog = true`. See useHistoryLog.
`exportClipboard`	`boolean`	Read-write. Indicates whether to retain Adobe Photoshop CS3 contents on the clipboard after you exit the application.
`fontPreviewSize`	`FontPreviewType`	Read-write. Indicates whether to show font previews in the type tool font menus.
`fullSizePreview`	`boolean`	Read-write. (Mac only.) Indicates whether to show image preview as a full size image or thumbnail**.**
`gamutWarningOpacity`	`number (double)`	Read-write. (0 - 100 as percent).
`gridSize`	`GridSize`	Read-write. The size to use for squares in the grid.
`gridStyle`	`GridLineStyle`	Read-write. The formatting style for non-printing grid lines.
`gridSubDivisions`	`number (long)`	Read-write. (1 - 100)
`guideStyle`	`GuideLineStyle`	Read-write. The formatting style for non-printing guide lines.
`iconPreview`	`boolean`	Read-write. (Mac only.)
`imageCacheLevels`	`number (long)`	Read-write. The number of images to hold in the cache (1 - 8).
`imagePreviews`	`SaveBehavior`	Read-write. The behavior mode to use when saving files.
`interpolation`	`ResampleMethod`	Read-write. The method to use to assign color values to any new pixels created when an image is resampled or resized.

Property	Value Type	What it is (Continued)
`keyboardZoomResizesWindows`	`boolean`	Read-write. Indicates whether to automatically resize the window when zooming in or out using keyboard shortcuts.
`macOSThumbnail`	`boolean`	Read-write. (Mac only.) Indicates whether to create a thumbnail when saving the image.
`maximizeCompatibility`	`QueryStateType`	Read-write. The behavior to use to check whether to maximize compatibility when opening Adobe Photoshop CS3 (PSD) files .
`maxRAMuse`	`number (long)`	Read-write. The maximum percentage of available RAM used by Adobe Photoshop CS3 (5 - 100).
`nonLinearHistory`	`boolean`	Read-write. Indicates whether to allow non-linear history.
`numberofHistoryStates`	`number (long)`	Read-write. The number of history states to preserve (1 - 100).
`otherCursors`	`OtherPaintingCursors`	Read-write. The type of pointer to use.
`paintingCursors`	`PaintingCursors`	Read-write. The type of pointer to use.
`parent`	`object (Application)`	Read-write. The `preferences` object's container.
`pixelDoubling`	`boolean`	Read-write. Indicates whether to halve the resolution or (double the size of pixels) to make previews display more quickly.
`pointSize`	`PointType`	Read-write. The point/pica size.
`recentFileListLength`	`number (long)`	Read-write. The number of items in the recent file list (0 - 30).
`rulerUnits`	`Units`	Read-write. The unit the scripting system will use when receiving and returning values.
`saveLogItems`	`SaveLogItemsType`	Read-write. The options for saving the history items.
`saveLogItemsFile`	`File`	Read-write. The path to the history log file. **Note:** For information about the File object, see the *JavaScript Tools Guide*

Property	Value Type	What it is (Continued)
savePaletteLocations	boolean	Read-write. Indicates whether to make new palette locations the default location.
showAsianTextOptions	boolean	Read-write. Indicates whether to display Asian text options in the Paragraph palette.
showEnglishFontNames	boolean	Read-write. Indicates whether to list Asian font names in English.
showSliceNumber	boolean	Read-write. Indicates whether to display slice numbers in the document window when using the Slice tool.
showToolTips	boolean	Read-write. Indicates whether to show pop up definitions on mouse over.
smartQuotes	boolean	Read-write. Indicates whether to use curly or straight quote marks.
typename	string	Read-only. The class name of the referenced preferences object.
typeUnits	TypeUnits	Read-write. The unit type-size that the numeric inputs are assumed to represent.
useAdditionalPluginFolder	boolean	Read-write. Indicates whether to use an additional folder for compatible plug-ins stored with a different application.
useHistoryLog	boolean	Read-write. Indicates whether to create a log file for history states.
useLowerCaseExtension	boolean	Read-write. Indicates whether the file extension should be lowercase.
useShiftKeyForToolSwitch	boolean	Read-write. Indicates whether to enable cycling through a set of hidden tools.
useVideoAlpha	boolean	Read-write. Indicates whether to enable Adobe Photoshop CS3 to send transparency information to your computer’s video board. (Requires hardware support.)
windowsThumbnail	boolean	Read-write. (Requires hardware support.) Indicates whether to create a thumbnail when saving the image on Windows.

PresentationOptions

Options that can be specified for Adobe PDF presentations.

Properties

Property	Value Type	What it is
`autoAdvance`	`boolean`	Read-write. Indicates whether to auto advance images when when viewing the presentation (default: `true`). **Note:** Valid only when `presentation = true`. See presentation.
`includeFilename`	`boolean`	Read-write. Indicates whether to include the file name for the image (default: `false`).
`interval`	`number (long)`	Read-write. The time in seconds before the view is auto advanced (1 - 60; default: `5`). **Note:** Valid only when `AutoAdvance = true`. See autoAdvance.
`loop`	`boolean`	Read-write. Indicates whether to begin the presentation again after the last page (default: `false`). **Note:** Valid only when `autoAdvance = true`. See autoAdvance.
`magnification`	`MagnificationType`	Read-write. The magnification type to use when viewing the image.
`pDFFileOptions`	`PDFSaveOptions`	Read-write. Options to use when creating the PDF file.
`presentation`	`boolean`	Read-write. Indicates whether the output will be a presentation (default: `false`); when `false`, the output is a Multi-Page document.
`transition`	`TransitionType`	Read-write. The transition from one image to the next (default: `TransitionType.NONE`). **Note:** Valid only when `autoAdvance = true`. See autoAdvance.
`typename`	`string`	Read-only. The class name of the referenced `PresentationOptions` object.

RawFormatOpenOptions

Options that can be specified when opening a document in RAW format.

Properties

Property	Value Type	What it is
`bitsPerChannel`	`number (long)`	Read-write. The number of bits for each channel. **Note:** The only valid values are `bitsPerChannel = BitsPerChannelType.EIGHT` or `bitsPerChannel = BitsPerChannelType.SIXTEEN`.
`byteOrder`	`ByteOrder`	Read-write. The order in which bytes will be read. **Note:** Valid only when `bitsPerChannel = BitsPerChannelType.SIXTEEN`. See bitsPerChannel.
`channelNumber`	`number (long)`	Read-write. The number of channels in the image (1 - 56). **Note:** The value of `channelNumber` cannot exceed the number of channels in the image. When `bitsPerChannel = BitsPerChannelType.SIXTEEN`, only the following values are valid: 1, 3, or 4. See bitsPerChannel.
`headerSize`	`number (long)`	Read-write. The number of bytes of information that will appear in the file before actual image information begins; that is, the number of zeroes inserted at the beginning of the file as placeholders (0 - 1919999).
`height`	`number (long)`	Read-write. The height of the image (in pixels).
`interleaveChannels`	`boolean`	Read-write. Indicates whether to store color values sequentially.
`retainHeader`	`boolean`	Read-write. Indicates whether to retain the header when saving. **Note:** Valid only when headerSize is 1 or greater.
`typename`	`string`	Read-only. The class name of the referenced `RawFormatOpenOptions` object.
`width`	`number (long)`	Read-write. The image width in pixels.

RawSaveOptions

Options that can be specified when saving a document in RAW format.

Properties

Property	Value Type	What it is
`alphaChannels`	`boolean`	Read-write. Indicates whether alpha channels should be saved.
`spotColors`	`boolean`	Read-write. Indicates whether the spot colors should be saved.
`typename`	`string`	Read-only. The class name of the referenced `RawSaveOptions` object.

RGBColor

The definition of a color in RGB color mode.

Properties

Property	Value Type	What it is
blue	`number (double)`	Read-write. The blue color value (0.0 - 255.0; default: 255.0).
green	`number (double)`	Read-write. The green color value (0.0 - 255.0; default: 255.0).
hexValue	`string`	Read-write. The hex representation of the color.
red	`number (double)`	Read-write. The red color value (0.0 - 255.0; default: 255.0).
typename	`string`	Read-only. The class name of the referenced `RGBColor` object.

Selection

The selected area of a document or layer.

Note: Many of the properties and methods of `Selection` use the `UnitValue` type. For information about this type, see the *JavaScript Tools Guide*.

Note: Because the `Selection` class is a property of the Document object, you use the property name, `selection`, rather than the class name, `Selection`, in your code, as in the following example:

```
checkersDoc.selection.fill(app.foregroundColor)
```

Properties

Property	Value Type	What it is
bounds	array of UnitValue	Read-only. The bounding rectangle of the entire selection.
parent	object (Document)	Read-only. The object's container.
solid	boolean	Read-only. Indicates if the bounding rectangle is a solid.
typename	string	Read-only. The class name of the referenced `selection` object.

Methods

Method	Parameter Type	Returns	What it does
clear ()			Clears the selection and does not copy it to the clipboard.
contract (by)	UnitValue		Contracts the selection by the specified amount.
copy ([merge])	boolean		Copies the selection to the clipboard. When the optional argument is used and set to `true`, a merged copy is performed (all visible layers in the selection are copied).
cut ()			Clears the current selection and copies it to the clipboard.
deselect ()			Deselects the current selection.
expand (by)	UnitValue		Expands the selection by the specified amount.
feather (by)	UnitValue		Feathers the edges of the selection by the specified amount.

Method	Parameter Type	Returns	What it does (Continued)
fill (filltype [, mode] [, opacity] [, preserveTransparency])	Object (SolidColor or HistoryState) ColorBlendMode number (long) boolean		Fills the selection (`opacity`: 1 - 100 as percent).
grow (tolerance, antiAlias)	number (long) boolean		Grows the selection to include all adjacent pixels falling within the specified tolerance range.
invert ()			Inverts the selection (deselects the selection and selects the rest of the layer or document). **Note:** To flip the selection shape, see rotate.
load (from [, combination] [, inverting])	Channel SelectionType boolean		Loads the selection from the specified channel.
makeWorkPath ([tolerance])	number (double)		Makes this selection item the work path for this document.
resize ([horizontal] [, vertical] [, anchor])	number (double) number (double) AnchorPosition		Resizes the selected area to the specified dimensions and anchor position.
resizeBoundary ([horizontal] [, vertical] [, anchor])	number (double) number (double) AnchorPosition		Changes the size of the selection to the specified dimensions around the specified anchor.
rotate (angle [, anchor])	number (double) AnchorPosition		Rotates the selection by the specified amount around the specified anchor point.
rotateBoundary (angle [, anchor])	number (double) AnchorPosition		Rotates the boundary of the selection around the specified anchor.
select (region [, type] [, feather] [, antiAlias])	array of points: Array (Array (x,y),...) SelectionType number (double) boolean		Selects the specified region.
selectAll ()			Selects the entire layer.

Method	Parameter Type	Returns	What it does (Continued)
selectBorder (width)	UnitValue		Selects the selection border only (in the specified width); subsequent actions do not affect the selected area within the borders.
similar (tolerance, antiAlias)	number (long) boolean		Grows the selection to include pixels throughout the image falling within the tolerance range.
smooth (radius)	number (long)		Cleans up stray pixels left inside or outside a color-based selection (within the radius specified in pixels).
store (into [, combination])	Channel SelectionType		Saves the selection as a channel.
stroke (strokeColor, width [, location] [, mode] [, opacity] [, preserveTransparency])	Object (SolidColor) number (long) StrokeLocation ColorBlendMode number (long) boolean		Strokes the selection border (opacity: 1 - 100 as percent).
translate ([deltaX] [, deltaY])	UnitValue UnitValue		Moves the entire selection relative to its current position.
translateBoundary ([deltaX] [, deltaY])	UnitValue UnitValue		Moves the selection relative to its current position.

Sample Script

The following script creates a checkerboard using the following steps:

- Create an 800 x 800 pixel document.
- Divide the entire document into 100 x 100 pixel squares.
- Select every other square in the first row, then shift the selection criteria to select the alternate squares in the following row. Repeat until every other square in the document is selected.
- Fill the selected squares with the foreground color from the palette.
- Invert the selection and fill the newly selected squares with the background color from the palette.
- Deselect the squares to remove the selection outlines (the “marching ants").

Selection.jsx

```
// Save the current preferences
var startRulerUnits = app.preferences.rulerUnits
var startTypeUnits = app.preferences.typeUnits
var startDisplayDialogs = app.displayDialogs

// Set Adobe Photoshop CS3 to use pixels and display no dialogs
app.preferences.rulerUnits = Units.PIXELS
app.preferences.typeUnits = TypeUnits.PIXELS
app.displayDialogs = DialogModes.NO

//Close all the open documents
while (app.documents.length) {
   app.activeDocument.close()
}

//Create variables for the 800 pixel board divided in even 100 x 100 squares
var docSize = 800
var cells = 8
var cellSize = docSize / cells

// create a new document
var checkersDoc = app.documents.add(docSize, docSize, 72, "Checkers")

// Create a variable to use for selecting the checker board
// That allows me to shift the selection one square to the right
//on every other row, and then shift back for the rows in between.
var shiftIt = true

// loop through vertically to create the first row
for (var v = 0; v < docSize; v += cellSize) {

   // Switch the shift for a new row
   shiftIt = !shiftIt

   // loop through horizontally
   for (var h = 0; h < docSize; h += (cellSize * 2)) {

          // push over the cellSize to start with only
          if (shiftIt && h == 0) {
             h += cellSize
      }

          // Select a square
          selRegion = Array(Array(h, v),
                      Array(h + cellSize, v),
                      Array(h + cellSize, v + cellSize),
                      Array(h, v + cellSize),
                      Array(h, v))

          // In the first ineration of the loop, start the selection
          //In subsequent iterations, use the EXTEND constant value
          //of the select() method to add to the selection (in the loop’s else clause)
          if (h == 0 && v == 0) {
             checkersDoc.selection.select(selRegion)
          } else {
             checkersDoc.selection.select(selRegion, SelectionType.EXTEND)
      }

          // turn this off for faster execution
          // turn this on for debugging
          WaitForRedraw()
   }
}
```

```
// Fill the current selection with the foreground color
checkersDoc.selection.fill(app.foregroundColor)

//Invert the selection
checkersDoc.selection.invert()

// Fill the new selection with the background color
checkersDoc.selection.fill(app.backgroundColor)

// Clear the selection to get rid of the non-printing borders
checkersDoc.selection.deselect()

// Reset the application preferences
app.preferences.rulerUnits = startRulerUnits
app.preferences.typeUnits = startTypeUnits
app.displayDialogs = startDisplayDialogs

// A helper function for debugging
// It also helps the user see what is going on
// if you turn it off for this example you
// get a flashing cursor for a number (long) time
function WaitForRedraw()
{
   var eventWait = charIDToTypeID("Wait")
   var enumRedrawComplete = charIDToTypeID("RdCm")
   var typeState = charIDToTypeID("Stte")
   var keyState = charIDToTypeID("Stte")

   var desc = new ActionDescriptor()

   desc.putEnumerated(keyState, typeState, enumRedrawComplete)

   executeAction(eventWait, desc, DialogModes.NO)
}
```

SGIRGBSaveOptions

Options that can be specified when saving a document in SGIRGB format.

Note: The SGIRGB format is not installed automatically with Adobe Photoshop CS3.

Properties

Property	Value Type	What it is
`alphaChannels`	`boolean`	Read-write. Indicates whether to save the alpha channels.
`spotColors`	`boolean`	Read-write. Indicates whether to save the spot colors.
`typename`	`string`	Read-only. The class name of the referenced `SGIRGBSaveOptions` object.

SolidColor

A color definition used in the document.

Properties

Property	Value Type	What it is
cmyk	CMYKColor	Read-write. The CMYK color mode.
gray	GrayColor	Read-write. The Grayscale color mode.
hsb	HSBColor	Read-write. The HSB color mode.
lab	LabColor	Read-write. The LAB color mode.
model	ColorModel	Read-write. The color model.
nearestWebColor	RGBColor	Read-only. The nearest web color to the current color.
rgb	RGBColor	Read-write. The RGB color mode.
typename	string	Read-only. The class name of the referenced SolidColor object.

Methods

Method	Parameter Type	Returns	What it does
isEqual (color)	SolidColor	boolean	Indicates whether the SolidColor object is visually equal to the specified color.

SubPathInfo

An array of `PathPointInfo` objects that describes a straight or curved segment of a path.

You add sub path information to a path by passing a `SubPathInfo` object into the add method of the PathItems class. This method creates the SubPathItem objects associated with each SubPathInfo object, and returns a PathItem object that is the path represented by all the sub paths.

Properties

Property	Value Type	What it is
`closed`	`boolean`	Read-write. Indicates whether the path describes an enclosed area.
`entireSubPath`	`Array (PathPoint objects)`	Read-write.
`operation`	`ShapeOperation`	Read-write. The sub path's operation on other sub paths.
`typename`	`string`	Read-only. The class name of the referenced `SubPathInfo` object.

SubPathItem

Information about a path.

Note: You do not use the `SubPathItem` object to create a path. Rather, you create path segments using the SubPathInfo object. Use the `SubPathItem` object to retrieve information about a path. (Note that all of the `SubPathItem` object's properties are *Read-only*.)

Properties

Property	Value Type	What it is
`closed`	`boolean`	Read-only. Indicates whether the path is closed.
`operation`	`ShapeOperation`	Read-only. The sub path operation on other sub paths.
`parent`	`object (PathItem)`	Read-only. The object's container.
`pathPoints`	`PathPoints`	Read-only. The `PathPoints` collection.
`typename`	`string`	Read-only. The class name of the referenced `SubPathItem` object.

SubPathItems

A collection of SubPathItem objects. See SubPathItem.

Properties

Property	Value Type	What it is
length	number (long)	Read-only. The number of elements in the SubPathItems collection.
parent	object (PathItem)	Read-only. The SubPathItems object's container.
typename	string	Read-only. The class name of the referenced SubPathItems object.

Methods

Method	Parameter type	Returns	What it does
index (itemKey)	number	SubPathItem	Gets an element from the SubPathItems collection.

TargaSaveOptions

Options that can be set when saving a document in TGA (Targa) format.

Properties

Property	Value Type	What it is
alphaChannels	boolean	Read-write. Indicates whether to save the alpha channels.
resolution	TargaBitsPerPixels	Read-write. The number of bits per pixel (default: TargaBitsPerPixels.TWENTYFOUR).
rleCompression	boolean	Read-write. Indicates whether RLE compression should be used (default: true).
typename	string	Read-only. The class name of the referenced TargaSaveOptions object.

TextFont

Details about a font in the `TextFonts` collection. See TextFonts for more information on the `TextFonts` collection.

Properties

Property	Value Type	What it is
`family`	`string`	Read-only. The font family.
`name`	`string`	Read-only. The name of the font.
`parent`	`object (Application)`	Read-only. The object's container.
`postScriptName`	`string`	Read-only. The PostScript name of the font.
`style`	`string`	Read-only. The font style.
`typename`	`string`	Read-only. The class name of the referenced `TextFont` object.

TextFonts

The collection of fonts available on your computer.

Note: The `TextFonts` class corresponds to the `fonts` property of the `TextFonts` object. In a script, you use the property name `fonts` , rather than the class name `TextFonts`, to refer to a `TextFonts` object. The following example uses the length property to determine, and then display, the number of `TextFonts` installed on the machine.

- Correct:

```
alert(app.fonts.length)
```

- Incorrect:

```
alert(app.TextFonts.length)
```

See `Application`, specifically the `fonts` property, for more information.

Properties

Property	Value Type	What it is
`length`	`number (long)`	Read-only. The number of elements in the `TextFonts` collection.
`parent`	`object (Application)`	Read-only. The object's container.
`typename`	`string`	Read-only. The class name of the referenced `TextFonts` object.

Methods

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

TextItem

The text in an `artLayer` object whose `kind` property is `LayerKind.TEXT`. See ArtLayer, specifically the `kind` property, for more information.

Note: Many of the properties i n the `TextItem` class use the `UnitValue` type. For information about this type, see the *JavaScript Tools Guide*.

Note: Because the `TextItem` class is a property of the `ArtLayer` class, you use the property name, `textItem`, rather than the class name, `TextItem`, in your code. For example:

```
myLayers[i].textItem.contents = "Layer in " + textArray[i] + " Set Inside "
```

Properties

Property	Value Type	What it is
`alternateLigatures`	`boolean`	Read-write. Indicates whether to use alternate ligatures. **Note:** Alternate ligatures are the same as Discretionary Ligatures. Please refer to Adobe Photoshop CS3 Help for more information.
`antiAliasMethod`	`AntiAlias`	Read-write. The method of anti aliasing to use.
`autoKerning`	`AutoKernType`	Read-write. The auto kerning option to use.
`autoLeadingAmount`	`number (double)`	Read-write. The percentage to use for auto (default) leading (0.01 - 5000.00 in points). **Note:** Valid only when `useAutoLeading = true`. See useAutoLeading.
`baselineShift`	`UnitValue`	Read-write. The unit value to use in the baseline offset of text.
`capitalization`	`TextCase`	Read-write. The text case.
`color`	`SolidColor`	Read-write. The text color.
`contents`	`string`	Read-write. The actual text in the layer.
`desiredGlyphScaling`	`number (double)`	Read-write. The desired amount (percentage) to scale the horizontal size of the text letters (50 - 200; at 100, the width of characters is not scaled). **Note:** Valid only when `justification = Justification.CENTERJUSTIFIED`; `Justification.FULLYJUSTIFIED`; `Justification.LEFTJUSTIFIED`; or `Justification.RIGHTJUSTIFIED`. See justification. The following values are also required: minimumGlyphScaling and maximumGlyphScaling.

Property	Value Type	What it is (Continued)
`desiredLetterScaling` **Note:** 'Letter Scaling' is basically equivalent to 'Letter Spacing' in the Adobe Photoshop CS3 application Justification dialog (Select Justification on the Paragraphs palette menu).	`number (double)`	Read-write. The amount of space between letters (100 - 500; at 0, no space is added between letters). **Note:** Valid only when `justification = Justification.CENTERJUSTIFIED`; `Justification.FULLYJUSTIFIED`; `Justification.LEFTJUSTIFIED`; or `Justification.RIGHTJUSTIFIED`. See justification. The following values are also required: minimumLetterScaling and maximumLetterScaling.
`desiredWordScaling` **Note:** 'Word Scaling' is basically equivalent to 'Word Spacing' in the Adobe Photoshop CS3 application Justification dialog (Select Justification on the Paragraphs palette menu).	`number (double)`	Read-write. The amount (percentage) of space between words (0 -1000; at 100, no additional space is added between words). **Note:** Valid only when `justification = Justification.CENTERJUSTIFIED`; `Justification.FULLYJUSTIFIED`; `Justification.LEFTJUSTIFIED`; or `Justification.RIGHTJUSTIFIED`. See justification. The following values are also required: minimumWordScaling and maximumWordScaling.
`direction`	`Direction`	Read-write. The text orientation.
`fauxBold`	`boolean`	Read-write. Indicates whether to use faux bold (default: `false`). **Note:** Using `fauxBold.true` is equivalent to selecting text and clicking the Faux Bold button in the Character palette.
`fauxItalic`	`boolean`	Read-write. Indicates whether to use faux italic (default: `false`). **Note:** Using `fauxItalic.true` is equivalent to selecting text and clicking the Faux Italic button in the Character palette.
`firstLineIndent`	`UnitValue`	Read-write. The amount (unit value) to indent the first line of paragraphs (-1296 - 1296).
`font`	`string`	Read-write. The text face of the character.
`hangingPunctuation`	`boolean`	Read-write. Indicates whether to use roman Hanging Punctuation.

Property	Value Type	What it is (Continued)
height	UnitValue	Read-write. The height of the bounding box (unit value) for paragraph text. **Note:** Valid only when `kind = TextType.PARAGRAPHTEXT`. See kind.
horizontalScale	number (long)	Read-write. Character scaling (horizontal) in proportion to vertical scale (0 - 1000 in percent). See verticalScale.
hyphenateAfterFirst	number (long)	Read-write. The number of letters after which hyphenation in word wrap is allowed (1 - 15).
hyphenateBeforeLast	number (long)	Read-write. The number of letters before which hyphenation in word wrap is allowed (1 - 15).
hyphenateCapitalWords	boolean	Read-write. Indicates whether to allow hyphenation in word wrap of capitalized words.
hyphenateWordsLongerThan	number (long)	Read-write. The minimum number of letters a word must have in order for hyphenation in word wrap to be allowed (2 - 25).
hyphenation	boolean	Read-write. Indicates whether to use hyphenation in word wrap.
hyphenationZone	UnitValue	Read-write. The distance at the end of a line that will cause a word to break in unjustified type (0 - 720 pica).
hyphenLimit	number (long)	Read-write. The maximum number of consecutive lines that can end with a hyphenated word.
justification	Justification	Read-write. The paragraph justification.
kind	TextType	Read-write. The text-wrap type.
language	Language	Read-write. The language to use.
leading	UnitValue	Read-write. The leading amount (unit value).
leftIndent	UnitValue	Read-write. The amount (unit value) of space to indent text from the left (-1296 - 1296).
ligatures	boolean	Read-write. Indicates whether to use ligatures.

Property	Value Type	What it is (Continued)
`maximumGlyphScaling`	`number (double)`	Read-write. The maximum amount (percentage) to scale the horizontal size of the text letters (50 - 200; at 100, the width of characters is not scaled). **Note:** Valid only when `justification =` `Justification.CENTERJUSTIFIED`; `Justification.FULLYJUSTIFIED`; `Justification.LEFTJUSTIFIED`; or `Justification.RIGHTJUSTIFIED`. See justification. The following values are also required: minimumGlyphScaling and desiredGlyphScaling.
`maximumLetterScaling` **Note:** 'Letter Scaling' is basically equivalent to 'Letter Spacing' in the Adobe Photoshop CS3 application Justification dialog (Select Justification on the Paragraphs palette menu).	`number (double)`	Read-write. The maximum amount of space to allow between letters (100 - 500; at 0, no space is added between letters). **Note:** Valid only when `justification =` `Justification.CENTERJUSTIFIED`; `Justification.FULLYJUSTIFIED`; `Justification.LEFTJUSTIFIED`; or `Justification.RIGHTJUSTIFIED`. See justification. The following values are also required: minimumLetterScaling and desiredLetterScaling.
`maximumWordScaling` **Note:** 'Word Scaling' is basically equivalent to 'Word Spacing' in the Adobe Photoshop CS3 application Justification dialog (Select Justification on the Paragraphs palette menu).	`number (double)`	Read-write. The maximum amount (percentage) of space to allow between words (0 -1000; at 100, no additional space is added between words). **Note:** Valid only when `justification =` `Justification.CENTERJUSTIFIED`; `Justification.FULLYJUSTIFIED`; `Justification.LEFTJUSTIFIED`; or `Justification.RIGHTJUSTIFIED`. See justification. The following values are also required: minimumWordScaling and desiredWordScaling.

Property	Value Type	What it is (Continued)
`minimumGlyphScaling`	`number (double)`	Read-write. The minimum amount (percentage) to scale the horizontal size of the text letters (50 - 200; at 100, the width of characters is not scaled). **Note:** Valid only when `justification = Justification.CENTERJUSTIFIED`; `Justification.FULLYJUSTIFIED`; `Justification.LEFTJUSTIFIED`; or `Justification.RIGHTJUSTIFIED`. See justification. The following values are also required: minimumGlyphScaling and desiredGlyphScaling.
`minimumLetterScaling` **Note:** 'Letter Scaling' is basically equivalent to 'Letter Spacing' in the Adobe Photoshop CS3 application Justification dialog (Select Justification on the Paragraphs palette menu).	`number (double)`	Read-write. The minimum amount (percentage) of space between letters (100 - 500; at 0, no space is removed between letters). **Note:** Valid only when `justification = Justification.CENTERJUSTIFIED`; `Justification.FULLYJUSTIFIED`; `Justification.LEFTJUSTIFIED`; or `Justification.RIGHTJUSTIFIED`. See justification. The following values are also required: maximumLetterScaling and desiredLetterScaling.
`minimumWordScaling` **Note:** 'Word Scaling' is basically equivalent to 'Word Spacing' in the Adobe Photoshop CS3 application Justification dialog (Select Justification on the Paragraphs palette menu).	`number (double)`	Read-write. The minimum amount (percentage) of space between words (0 -1000; at 100, no space is removed between words). **Note:** Valid only when `justification = Justification.CENTERJUSTIFIED`; `Justification.FULLYJUSTIFIED`; `Justification.LEFTJUSTIFIED`; or `Justification.RIGHTJUSTIFIED`. See justification. The following values are also required: maximumWordScaling and desiredWordScaling.
`noBreak`	`boolean`	Read-write. Indicates whether to allow words to break at the end of a line. **Tip:** When enacted on large amounts of consecutive characters, `noBreak = true` can prevent word wrap and thus may prevent some text from appearing on the screen.
`oldStyle`	`boolean`	Read-write. Indicates whether to use old style type.

Property	Value Type	What it is (Continued)
parent	object (ArtLayer)	Read-write. The TextItem object's container.
position	array (UnitValue)	Read-write. The position of origin for the text. The array must contain two values (unit value). **Tip:** Setting the position property is basically equivalent to clicking the text tool at a point in the document to create the point of origin for text.
rightIndent	UnitValue	Read-write. The amount of space (unit value) to indent text from the right (-1296 - 1296).
size	number (double)	Read-write. The font size in points.
spaceAfter	UnitValue	Read-write. The amount of space (unit value) to use after each paragraph (-1296 - 1296).
spaceBefore	UnitValue	Read-write. The amount of space (unit value) to use before each paragraph (-1296 - 1296).
strikeThru	StrikeThruType	Read-write. The text strike through option to use.
textComposer	TextComposer	Read-write. The composition method to use to evaluate line breaks and optimize the specified hyphenation and justification options. **Note:** Valid only when kind = TextType.PARAGRAPHTEXT. See kind.
tracking	number (double)	Read-write. The amount of uniform spacing between multiple characters (-1000 - 10000). **Note:** Tracking units are 1/1000 of an em space. The width of an em space is relative to the current type size. In a 1-point font, 1 em equals 1 point; in a 10-point font, 1 em equals 10 points. So, for example, 100 units in a 10-point font are equivalent to 1 point.
typename	string	Read-only. The class name of the referenced textItem object.
underline	UnderlineType	Read-write. The text underlining options.
useAutoLeading	boolean	Read-write. Indicates whether to use a font's built-in leading information.
verticalScale	number (long)	Read-write. Character scaling (vertical) in proportion to horizontal scale (0 - 1000 in percent). See horizontalScale.
warpBend	number (double)	Read-write. The warp bend percentage (-100 - 100).
warpDirection	Direction	Read-write. The warp direction.

Property	Value Type	What it is (Continued)
`warpHorizontalDistortion`	`number (double)`	Read-write. The horizontal distortion (as percentage) of the warp (-100 - 100).
`warpStyle`	`WarpStyle`	Read-write. The style of warp to use.
`warpVerticalDistortion`	`number (double)`	Read-write. The vertical distortion (as percentage) of the warp (-100 - 100).
`width`	`UnitValue`	Read-write. The width of the bounding box (unit value) for paragraph text. **Note:** Valid only when `kind = TextType.PARAGRAPHTEXT`. See kind.

Methods

Method	Parameter Type	Returns	What it does
`convertToShape` `()`			Converts the text item and its containing layer to a fill layer with the text changed to a clipping path.
`createPath` `()`			Creates a clipping path from the outlines of the actual text items (such as letters or words).

TiffSaveOptions

Options that can be specified when saving a document in TIFF format.

Properties

Property	Value Type	What it is
`alphaChannels`	`boolean`	Read-write. Indicates whether to save the alpha channels.
`annotations`	`boolean`	Read-write. Indicates whether to save the annotations.
`byteOrder`	`ByteOrder`	Read-write. The order in which the document's bytes will be read. (The default is `ByteOrder.MACOS` when running on MacOS and `ByteOrder.IBM` when running on a PC.)
`embedColorProfile`	`boolean`	Read-write. Indicates whether to embed the color profile in the document.
`imageCompression`	`TIFFEncoding`	Read-write. The compression type (default: `TIFFEncoding.NONE`).
`interleaveChannels`	`boolean`	Read-write. Indicates whether the channels in the image will be interleaved.
`jpegQuality`	`number (long)`	Read-write. The quality of the produced image (0 - 12), which is inversely proportionate to the amount of JPEG compression. **Note:** Valid only when `imageCompression = TIFFEncoding.JPEG`.
`layerCompression`	`LayerCompression`	Read-write. The method of compression to use when saving layers (as opposed to saving composite data). **Note:** Valid only when `layers = true`. See layers
`layers`	`boolean`	Read-write. Indicates whether to save the layers.
`saveImagePyramid`	`boolean`	Read-write. Indicates whether to preserve multi-resolution information (default: `false`).
`spotColors`	`boolean`	Read-write. Indicates whether to save the spot colors.
`transparency`	`boolean`	Read-write. Indicates whether to save the transparency as an additional alpha channel when the file is opened in another application.
`typename`	`string`	Read-only. The class name of the referenced `TiffSaveOptions` object.

xmpMetadata

Camera RAW image file settings stored in an XMP file in the same folder as the RAW file with the same base name and an XMP extension.

Properties

Property	Value Type	What it is
`parent`	`object (Document)`	Read-only. The object's container.
`rawData`	`string`	Read-write. The raw XML form of file information.
`typename`	`string`	Read-only. The class name of the referenced `xmpMetadata` object.

3 Scripting Constants

This section lists and describes the enumerations defined for use with Adobe Photoshop CS3 JavaScript properties and methods.

Constant type	Values	What it means
AdjustmentReference	ABSOLUTE RELATIVE	Method to use for interpreting selective color adjustment specifications: ABSOLUTE = % of the whole; RELATIVE = % of the existing color amount.
AnchorPosition	BOTTOMCENTER BOTTOMLEFT BOTTOMRIGHT MIDDLECENTER MIDDLELEFT MIDDLERIGHT TOPCENTER TOPLEFT TOPRIGHT	The point on the object that does not move when the object is rotated or resized.
AntiAlias	CRISP NONE SHARP SMOOTH STRONG	Method to use to smooth edges by softening the color transition between edge pixels and background pixels.
AutoKernType	MANUAL METRICS OPTICAL	The type of kerning to use for characters.
BatchDestinationType	FOLDER NODESTINATION SAVEANDCLOSE	The destination, if any, for batch-processed files: FOLDER: Save modified versions of the files to a new location (leaving the originals unchanged); NODESTINATIONTYPE: Leave all files open; SAVEANDCLOSE: Save changes and close the files.
BitmapConversionType	CUSTOMPATTERN DIFFUSIONDITHER HALFTHRESHOLD HALFTONESCREEN PATTERNDITHER	Specifies the quality of an image you are converting to bitmap mode.
BitmapHalfToneType	CROSS DIAMOND ELLIPSE LINE ROUND SQUARE	Specifies the shape of the dots (ink deposits) in the halftone screen.
BitsPerChannelType	EIGHT ONE SIXTEEN THIRTYTWO	The number of bits per color channel.

Constant type	Values	What it means
BlendMode	COLORBLEND COLORBURN COLORDODGE DARKEN DIFFERENCE DISSOLVE EXCLUSION HARDLIGHT HARDMIX HUE LIGHTEN LINEARBURN LINEARDODGE LINEARLIGHT LUMINOSITY MULTIPLY NORMAL OVERLAY PASSTHROUGH PINLIGHT SATURATION SCREEN SOFTLIGHT VIVIDLIGHT	Controls how pixels in the image are blended.
BMPDepthType	BMP_A1R5G5B5 BMP_A4R4G4B4 BMP_A8R8G8B8 BMP_R5G6B5 BMP_R8G8B8 BMP_X1R5G5B5 BMP_X4R4G4B4 BMP_X8R8G8B8 EIGHT FOUR ONE SIXTEEN THIRTYTWO TWENTYFOUR	The number of bits per channel (also called pixel depth or color depth). The number selected indicates the exponent of 2. For example, a pixel with a bit-depth of EIGHT has 2^8, or 256, possible color values.
ByteOrder	IBM MACOS	The order in which bytes will be read.
CameraRAWSettingsType	CAMERA CUSTOM SELECTEDIMAGE	The default CameraRaw settings to use: the camera settings, custom settings, or the settings of the selected image.

CameraRAWSize	EXTRALARGE
	LARGE
	MAXIMUM
	MEDIUM
	MINIMUM
	SMALL
		The camera RAW size type options:
		EXTRALARGE=5120 x 4096
		LARGE=4096 x 2731
		MAXIMUM=6144 X 4096
		MEDIUM=3072 x 2048
		MINIMUM=1536 x 1024
		SMALL=2048 x 1365

Constant type	Values	What it means
ChangeMode	BITMAP CMYK GRAYSCALE INDEXEDCOLOR LAB MULTICHANNEL RGB	The type of color mode to use. **Note:** Color images must be changed to GRAYSCALE mode before you can change them to BITMAP mode.
ChannelType	COMPONENT MASKEDAREA SELECTEDAREA SPOTCOLOR	The type of channel: COMPONENT: related to document color mode MASKEDAREA: Alpha channel where color indicates masked area SELECTEDAREA: Alpha channel where color indicates selected are SPOTCOLOR: Alpha channel to store a spot color.
ColorBlendMode	BEHIND CLEAR COLOR COLORBURN COLORDODGE DARKEN DIFFERENCE DISSOLVE EXCLUSION HARDLIGHT HARDMIXBLEND HUE LIGHTEN LINEARBURN LINEARDODGE LINEARLIGHT LUMINOSITY MULTIPLY NORMAL OVERLAY PINLIGHT SATURATION SCREEN SOFTLIGHT VIVIDLIGHT	Color blend mode type.
ColorModel	CMYK GRAYSCALE HSB LAB NONE RGB	The color model to use.
ColorPicker	ADOBE APPLE PLUGIN WINDOWS	The color picker to use.
ColorProfile	CUSTOM NONE WORKING	The color profile type to use to manage this document.

Constant type	Values	What it means
ColorReductionType	ADAPTIVE BLACKWHITE CUSTOM GRAYSCALE MACINTOSH PERCEPTUAL RESTRICTIVE SELECTIVE WINDOWS	The color reduction algorithm option to use.
ColorSpaceType	ADOBERGB COLORMATCHRGB PROPHOTORGB SRGB	The type of color space to use.
CopyrightedType	COPYRIGHTEDWORK PUBLICDOMAIN UNMARKED	The copyright status of the document.
CreateFields	DUPLICATION INTERPOLATION	The method to use for creating fields.
CropToType	ARTBOX BLEEDBOX BOUNDINGBOX CROPBOX MEDIABOX TRIMBOX	The style to use when cropping a page.
DCSType	COLORCOMPOSITE GRAYSCALECOMPOSITE NOCOMPOSITE	The DCS format to use: COLORCOMPOSITE: Creates a color composite file in addition to DCS files; GRAYSCALECOMPOSITE: Creates a grayscale composite file in addition to DCS files; NOCOMPOSITE: Does not create a composite file.
DepthMapSource	IMAGEHIGHLIGHT LAYERMASK NONE TRANSPARENCYCHANNEL	The source to use for the depth map.
DescValueType	ALIASTYPE BOOLEANTYPE CLASSTYPE DOUBLETYPE ENUMERATEDTYPE INTEGERTYPE LISTTYPE OBJECTTYPE RAWTYPE REFERENCETYPE STRINGTYPE UNITDOUBLE	The value type of an object.
DialogModes	ALL ERROR NO	Controls the type (mode) of dialogs Photoshop displays when running scripts.
Direction	HORIZONTAL VERTICAL	The orientation of the object.

Constant type	Values	What it means
`DisplacementMapType`	`STRETCHTOFIT` `TILE`	Describes how the displacement map fits the image if the image is not the same size as the map.

`Dither`	`DIFFUSION`
	`NOISE`
	`NONE`
	`PATTERN`
		The type of dithering to use.

`DocumentFill`	`BACKGROUNDCOLOR`
	`TRANSPARENT`
	`WHITE`
		The fill of the document.

`DocumentMode`	`BITMAP`
	`CMYK`
	`DUOTONE`
	`GRAYSCALE`
	`INDEXEDCOLOR`
	`LAB`
	`MULTICHANNEL`
	`RGB`
		The color mode of the open document.

`EditLogItemsType`	`CONCISE`
	`DETAILED`
	`SESSIONONLY`
		The history log edit options:
		`CONCISE`: Save a concise history log.
		`DETAILED`: Save a detailed history log.
		`SESSIONONLY`: Save history log only for the session.

`ElementPlacement`	`INSIDE`
	`PLACEATBEGINNING`
	`PLACEATEND`
	`PLACEBEFORE`
	`PLACEAFTER`
		The object’s position in the Layers palette.
		Note: Not all values are valid for all object types. Please refer to the object property definition in JavaScript Object Reference to make sure you are using a valid value.

`EliminateFields`	`EVENFIELDS` `ODDFIELDS`	The type of fields to eliminate.
`ExportType`	`ILLUSTRATORPATHS` `SAVEFORWEB`	The export options to use.

`Extension`	`LOWERCASE`
	`NONE`
	`UPPERCASE`
		The formatting of the extension in the filename.

Constant type	Values	What it means
FileNamingType	DDMM DDMMYY DOCUMENTNAMELOWER DOCUMENTNAMEMIXED DOCUMENTNAMEUPPER EXTENSIONLOWER EXTENSIONUPPER MMDD MMDDYY SERIALLETTERLOWER SERIALLETTERUPPER SERIALNUMBER1 SERIALNUMBER2 SERIALNUMBER3 SERIALNUMBER4 YYDDMM YYMMDD YYYYMMDD	File naming options for the batch command.
FontPreviewType	LARGE MEDIUM NONE SMALL	The type size to use for font previews in the type tool font menus.
ForcedColors	BLACKWHITE NONE PRIMARIES WEB	The type of colors to be forced (included) into the color table: BLACKWHITE: Pure black and pure white; NONE; PRIMARIES: Red, green, blue, cyan, magenta, yellow, black, and white; WEB: the 216 web-safe colors.
FormatOptions	OPTIMIZEDBASELINE PROGRESSIVE STANDARDBASELINE	The option with which to save a JPEG file: OPTIMIZEDBASELINE: Optimized color and a slightly reduced file size; PROGRESSIVE: Displays a series of increasingly detailed scans as the image downloads; STANDARDBASELINE: Format recognized by most web browsers.
GalleryConstrainType	CONSTRAINBOTH CONSTRAINHEIGHT CONSTRAINWIDTH	The type of proportions to constrain for images.
GalleryFontType	ARIAL COURIERNEW HELVETICA TIMESNEWROMAN	The fonts to use for the Web photo gallery captions and other text.
GallerySecurityTextColorType	BLACK CUSTOM WHITE	The color to use for text displayed over gallery images as an antitheft deterrent.
GallerySecurityTextPositionType	CENTERED LOWERLEFT LOWERRIGHT UPPERLEFT UPPERRIGHT	The position of the text displayed over gallery images as an antitheft deterrent.

Constant type	Values	What it means
GallerySecurityTextRotateType	CLOCKWISE45 CLOCKWISE90 COUNTERCLOCKWISE45 COUNTERCLOCKWISE90 ZERO	The orientation of the text displayed over gallery images as an antitheft deterrent.
GallerySecurityType	CAPTION COPYRIGHT CREDIT CUSTOMTEXT FILENAME NONE TITLE	The content to use for text displayed over gallery images as an antitheft deterrent. **Note:** All types draw from the image's file information except CUSTOMTEXT.
GalleryThumbSizeType	CUSTOM LARGE MEDIUM SMALL	The size of thumbnail images in the web photo gallery.
Geometry	HEPTAGON HEXAGON OCTAGON PENTAGON SQUARE TRIANGLE	Geometric options for shapes, such as the iris shape in the Lens Blur Filter.
GridLineStyle	DASHED DOTTED SOLID	The line style for the nonprinting grid displayed over images.
GridSize	LARGE MEDIUM NONE SMALL	The value of grid line spacing.
GuideLineStyle	DASHED SOLID	The line style for nonprinting guides displayed over images.
IllustratorPathType	ALLPATHS DOCUMENTBOUNDS NAMEDPATH	The paths to export.
Intent	ABSOLUTECOLORIMETRIC PERCEPTUAL RELATIVECOLORIMETRIC SATURATION	The rendering intent to use when converting from one color space to another.
JavaScriptExecutionMode	BEFORERUNNING NEVER ONRUNTIMEERROR	The debugger mode to use.
Justification	CENTER CENTERJUSTIFIED FULLYJUSTIFIED LEFT LEFTJUSTIFIED RIGHT RIGHTJUSTIFIED	The placement of paragraph text within the bounding box.

Constant type	Values	What it means

`Language`	`BRAZILLIANPORTUGUESE`
	`CANADIANFRENCH`
	`DANISH`
	`DUTCH`
	`ENGLISHUK`
	`ENGLISHUSA`
	`FINNISH`
	`FRENCH`
	`GERMAN`
	`ITALIAN`
	`NORWEGIAN`
	`NYNORSKNORWEGIAN`
	`OLDGERMAN`
	`PORTUGUESE`
	`SPANISH`
	`SWEDISH`
	`SWISSGERMAN`
		The language to use.

`LayerCompression`	`RLE` `ZIP`	Compression methods for data for pixels in layers.

`LayerKind`	`BRIGHTNESSCONTRAST`
	`CHANNELMIXER`
	`COLORBALANCE`
	`CURVES`
	`EXPOSURE`
	`GRADIENTFILL`
	`GRADIENTMAP`
	`HUESATURATION`
	`INVERSION`
	`LEVELS`
	`NORMAL`
	`PATTERNFILL`
	`PHOTOFILTER`
	`POSTERIZE`
	`SELECTIVECOLOR`
	`SMARTOBJECT`
	`SOLIDFILL`
	`TEXT`
	`THRESHOLD`
	`LAYER3D`
	`VIDEO`
		The kind of `artLayer` object.
		Note: You can create a text layer only from an empty art layer.

`LensType`	`MOVIEPRIME`
	`PRIME105`
	`PRIME35`
	`ZOOMLENS`
		The type of lens to use.

`MagnificationType`	`ACTUALSIZE` `FITPAGE`	The type of magnification to use when viewing an image.

`MatteType`	`BACKGROUND`
	`BLACK`
	`FOREGROUND`
	`NETSCAPE`
	`NONE`
	`SEMIGRAY`
	`WHITE`
		The color to use for matting.

`MeasurementRange`	`ALLMEASUREMENTS` `ACTIVEMEASUREMENTS`	The measurement to take action upon

Constant type	Values	What it means
MeasurementSource	MEASURESELECTION MEASURECOUNTTOOL MEASURERULERTOOL	The source for recording measurements
NewDocumentMode	BITMAP CMYK GRAYSCALE LAB RGB	The color profile to use for the document.
NoiseDistribution	GAUSSIAN UNIFORM	Distribution method to use when applying an Add Noise filter.
OffsetUndefinedAreas	REPEATEDGEPIXELS SETTOBACKGROUND WRAPAROUND	Method to use to fill the empty space left by offsetting a an image or selection.
OpenDocumentMode	CMYK GRAYSCALE LAB RGB	The color profile to use.
OpenDocumentType	ALIASPIX BMP CAMERARAW COMPUSERVEGIF DICOM ELECTRICIMAGE EPS EPSPICTPREVIEW EPSTIFFPREVIEW FILMSTRIP JPEG PCX PDF PHOTOCD PHOTOSHOP PHOTOSHOPDCS_1 PHOTOSHOPDCS_2 PHOTOSHOPEPS PHOTOSHOPPDF PICTFILEFORMAT PICTRESOURCEFORMAT PIXAR PNG PORTABLEBITMAP RAW SCITEXCT SGIRGB SOFTIMAGE TARGA TIFF WAVEFRONTRLA WIRELESSBITMAP	The format in which to open the document. **Note:** PHOTOCD is deprecated. Kodak PhotoCD is now found in the Goodies folder on the Adobe Photoshop CS3 Install DVD. **Note:** The DICOM option is for the Extended version only.
OperatingSystem	OS2 WINDOWS	The operating system.
Orientation	LANDSCAPE PORTRAIT	The page orientation.

Constant type	Values	What it means
OtherPaintingCursors	PRECISEOTHER STANDARDOTHER	The pointer for the following tools: Eraser, Pencil, Paintbrush, Healing Brush, Rubber Stamp, Pattern Stamp, Smudge, Blur, Sharpen, Dodge, Burn, Sponge.
PaintingCursors	BRUSHSIZE PRECISE STANDARD	The pointer for the following tools: Marquee, Lasso, Polygonal Lasso, Magic Wand, Crop, Slice, Patch Eyedropper, Pen, Gradient, Line, Paint Bucket, Magnetic Lasso, Magnetic Pen, Freeform Pen, Measure, Color Sampler.
Palette	EXACT LOCALADAPTIVE LOCALPERCEPTUAL LOCALSELECTIVE MACOSPALETTE MASTERADAPTIVE MASTERPERCEPTUAL MASTERSELECTIVE PREVIOUSPALETTE UNIFORM WEBPALETTE WINDOWSPALETTE	The palette type to use.
PathKind	CLIPPINGPATH NORMALPATH TEXTMASK VECTORMASK WORKPATH	The type of path.
PDFCompatibility	PDF13 PDF14 PDF15 PDF16	The PDF version to make the document compatible with.
PDFEncoding	JPEG JPEG2000HIGH JPEG2000LOSSLESS JPEG2000LOW JPEG2000MED JPEG2000MEDHIGH JPEG2000MEDLOW JPEGHIGH JPEGLOW JPEGMED JPEGMEDHIGH JPEGMEDLOW NONE PDFZIP PDFZIP4BIT	The type of compression to use when saving a document in PDF format.
PDFResample	NONE PDFAVERAGE PDFBICUBIC PDFSUBSAMPLE	The down sample method to use.

Constant type	Values	What it means
PDFStandard	NONE PDFX1A2001 PDFX1A2003 PDFX32002 PDFX32003	The PDF standard to make the document compatible with.
PhotoCDColorSpace	LAB16 LAB8 RGB16 RGB8	The color space to use when creating a Photo CD. **Note:** Deprecated for Adobe Photoshop CS3. Kodak PhotoCD is now found in the Goodies folder on the Adobe Photoshop CS3 Install DVD.

PhotoCDSize	EXTRALARGE
	LARGE
	MAXIMUM
	MEDIUM
	MINIMUM
	SMALL
		The pixel dimensions of the image.
		EXTRALARGE = 1024x1536
		LARGE = 512x768
		MAXIMUM = 2048x3072
		MEDIUM = 256x384
		MINIMUM = 64x96
		SMALL = 128x192
		Note: Deprecated for Adobe Photoshop CS3. Kodak PhotoCD is now found in the Goodies folder on the Adobe Photoshop CS3 Install DVD.

PICTBitsPerPixels	EIGHT FOUR SIXTEEN THIRTYTWO TWO	The number of bits per pixel to use when compression a PICT file. **Note:** Use 16 or 32 for RGB images; use 2, 4, or 8 for bitmap and grayscale images.
PICTCompression	JPEGHIGHPICT JPEGLOWPICT JPEGMAXIMUMPICT JPEGMEDIUMPICT NONE	The type of compression to use when saving an image as a PICT file.
PicturePackageTextType	CAPTION COPYRIGHT CREDIT FILENAME NONE ORIGIN USER	The function or meaning of text in a Picture Package.
PointKind	CORNERPOINT SMOOTHPOINT	The role a pathPoint plays in a pathItem.

PointType	POSTSCRIPT
	TRADITIONAL
		The kind of measurement to use for type points:
		POSTSCRIPT = 72 points/inch;
		TRADITIONAL = 72.27 points/inch.

Constant type	Values	What it means
`PolarConversionType`	`POLARTORECTANGULAR` `RECTANGULARTOPOLAR`	The method of polar distortion to use.

`Preview`	`EIGHTBITTIFF`
	`MACOSEIGHTBIT`
	`MACOSJPEG`
	`MACOSMONOCHROME`
	`MONOCHROMETIFF`
	`NONE`
		The type of image to use as a low-resolution preview in the destination application.

`PrintEncoding`	`ASCII`
	`BINARY`
	`JPEG`
		The type of encoding to use.

`PurgeTarget`	`ALLCACHES`
	`CLIPBOARDCACHE`
	`HISTORYCACHES`
	`UNDOCACHES`
		Cache to be targeted in a purge operation.

`QueryStateType`	`ALWAYS`
	`ASK`
	`NEVER`
		Permission state for queries.

`RadialBlurMethod`	`SPIN` `ZOOM`	The blur method to use.

`RadialBlurQuality`	`BEST`
	`DRAFT`
	`GOOD`
		The smoothness or graininess of the blurred image.

`RasterizeType`	`ENTIRELAYER`
	`FILLCONTENT`
	`LAYERCLIPPINGPATH`
	`LINKEDLAYERS`
	`SHAPE`
	`TEXTCONTENTS`
		The layer element to rasterize.

`ReferenceFormType`	`CLASSTYPE`
	`ENUMERATED`
	`IDENTIFIER`
	`INDEX`
	`NAME`
	`OFFSET`
	`PROPERTY`
		The type of an ActionReference object.

`ResampleMethod`	`BICUBIC`
	`BICUBICSHARPER`
	`BICUBICSMOOTHER`
	`BILINEAR`
	`NEARESTNEIGHBOR`
	`NONE`
		The method to use for image interpolation.

`ResetTarget`	`ALLTOOLS`
	`ALLWARNINGS`
	`EVERYTHING`
		The type of object or objects to reset to default settings.

`RippleSize`	`LARGE`
	`MEDIUM`
	`SMALL`
		The size of undulations to use.

`SaveBehavior`	`ALWAYSSAVE`
	`ASKWHENSAVING`
	`NEVERSAVE`
		The application’s behavior when a `save()` method is called.

Constant type	**Values**	**What it means**
SaveDocumentType	ALIASPIX BMP COMPUSERVEGIF ELECTRICIMAGE JPEG PCX PHOTOSHOP PHOTOSHOPDCS_1 PHOTOSHOPDCS_2 PHOTOSHOPEPS PHOTOSHOPPDF PICTFileFORMAT PICTRESOURCEFORMAT PIXAR PNG PORTABLEBITMAP RAW SCITEXCT SGIRGB SOFTIMAGE TARGA TIFF WAVEFRONTRLA WIRELESSBITMAP	The format in which to save a document. **Note:** The format property of the ExportOptionsSaveForWeb class uses only the following values: COMPUSERVEGIF, JPEG, PNG-8, PNG-24, and BMP. See ExportOptionsSaveForWeb.
SaveEncoding	ASCII BINARY JPEGHIGH JPEGLOW JPEGMAXIMUM JPEGMEDIUM	The type of encoding to use when saving a file.
SaveLogItemsType	LOGFILE LOGFILEANDMETADATA METADATA	The location of history log data.
SaveOptions	DONOTSAVECHANGES PROMPTTOSAVECHANGES SAVECHANGES	The 'save' method to use when closing a document.
SelectionType	DIMINISH EXTEND INTERSECT REPLACE	The selection behavior when a selection already exists: DIMINISH: Remove the selection from the already selected area; EXTEND: Add the selection to an already selected area; INTERSECT: Make the selection only the area where the new selection intersects the already selected area; REPLACE: Replace the selected area.
ShapeOperation	SHAPEADD SHAPEINTERSECT SHAPESUBTRACT SHAPEXOR	A subPathItem object's behavior when it intersects another subPathItem object.
SmartBlurMode	EDGEONLY NORMAL OVERLAYEDGE	The method to use for smart blurring: EDGEONLY, OVERLAYEDGES: Apply blur only to edges of color transitions; NORMAL: Apply blur to entire image.

Constant type	Values	What it means
SmartBlurQuality	HIGH LOW MEDIUM	The blur quality to use.
SourceSpaceType	DOCUMENT PROOF	The color space for source when printing.
SpherizeMode	HORIZONTAL NORMAL VERTICAL	The curve (or stretch shape) to use for the distortion.
StrikeThruType	STRIKEBOX STRIKEHEIGHT STRIKEOFF	The style of strikethrough to use.
StrokeLocation	CENTER INSIDE OUTSIDE	The placement of path or selection boundary strokes.
TargaBitsPerPixels	SIXTEEN THIRTYTWO TWENTYFOUR	The resolution to use when saving an image in Targa format.
TextCase	ALLCAPS NORMAL SMALLCAPS	The case usage for type.
TextComposer	ADOBEEVERYLINE ADOBESINGLELINE	The composition method to use to optimize the specified hyphenation and justification options.
TextType	PARAGRAPHTEXT POINTTEXT	The type of text: PARAGRAPHTEXT: Text that wraps within a bounding box; POINTTEXT: Text that does not wrap.
TextureType	BLOCKS CANVAS FILE FROSTED TINYLENS	The type of texture or glass surface image to load for a texturizer or glass filter.
TIFFEncoding	JPEG NONE TIFFLZW TIFFZIP	The encoding to use for TIFF files.

Constant type	Values	What it means
ToolType	ARTHISTORYBRUSH BACKGROUNDERASER BLUR BRUSH BURN CLONESTAMP COLORREPLACEMENTTOOL DODGE ERASER HEALINGBRUSH HISTORYBRUSH PATTERNSTAMP PENCIL SHARPEN SMUDGE SPONGE	The tool selection.
TransitionType	BLINDSHORIZONTAL BLINDSVERTICAL BOXIN BOXOUT DISSOLVE GLITTERDOWN GLITTERRIGHT GLITTERRIGHTDOWN NONE RANDOM SPLITHORIZONTALIN SPLITHORIZONTALOUT SPLITVERTICALIN SPLITVERTICALOUT WIPEDOWN WIPELEFT WIPERIGHT WIPEUP	The method to use to transition from one image to the next in a PDF presentation.
TrimType	BOTTOMRIGHT TOPLEFT TRANSPARENT	Type of pixels to trim around an image: BOTTOMRIGHT = bottom right pixel color; TOPLEFT = top left pixel color.
TypeUnits	MM PIXELS POINTS	The unit to use for measuring text characters.
UndefinedAreas	REPEATEDGEPIXELS WRAPAROUND	The method to use to treat undistorted areas or areas left blank in an image to which the a filter in the Distort category has been applied.
UnderlineType	UNDERLINELEFT UNDERLINEOFF UNDERLINERIGHT	The placement of text underlining. **Note:** UnderlineType.UNDELINELEFT and UnderlineType.UNDELINERIGHT are valid only when direction = Direction.VERTICAL.

Constant type	Values	What it means

`Units`	`CM`
	`INCHES`
	`MM`
	`PERCENT`
	`PICAS`
	`PIXELS`
	`POINTS`
		The measurement unit for type and ruler increments.

`Urgency`	`FOUR`
	`HIGH`
	`LOW`
	`NONE`
	`NORMAL`
	`SEVEN`
	`SIX`
	`THREE`
	`TWO`
		The editorial urgency of the artwork.

`WarpStyle`	`ARC`
	`ARCH`
	`ARCLOWER`
	`ARCUPPER`
	`BULGE`
	`FISH`
	`FISHEYE`
	`FLAG`
	`INFLATE`
	`NONE`
	`RISE`
	`SHELLLOWER`
	`SHELLUPPER`
	`SQUEEZE`
	`TWIST`
	`WAVE`
		The warp style to use.

`WaveType`	`SINE`
	`SQUARE`
	`TRIANGULAR`
		The type of wave to use.

`WhiteBalanceType`	`ASSHOT`
	`AUTO`
	`CLOUDY`
	`CUSTOM`
	`DAYLIGHT`
	`FLASH`
	`FLUORESCENT`
	`SHADE`
	`TUNGSTEN`
		The lighting conditions to use (affects color balance).

`ZigZagType`	`AROUNDCENTER`
	`OUTFROMCENTER`
	`PONDRIPPLES`
		The method of zigzagging to use.

4 JavaScript Resource

This section describes the JavaScript resource that enables your JavaScripts to behave like a plug-in. This includes:

- the ability to specify a menu the script appears in as a command,
- a terminology resource so the script can function with the Action Manager, which allows your script to record and be automated by scripting parameters,
- a category to enable ordering and grouping of commands within menus, and
- an enable string that indicates whether the command is enabled or disabled given a set of conditions.

JavaScript Resource Syntax

The JavaScript Resource has an HTML-style syntax, with each `<tag>` matched by a closing `</tag>`. This resource needs to appear within comments (`/* ... */`) and should be defined at the top of your script file (within the first 10,240 characters of the file.)

Tag	Description
`<javascriptresource>`	The resource definition tag.
`<name>`	The command name that appears in the Photoshop menu. If this tag is not provided in the resource, the name of the command in the menu defaults to the name of the script.
`<type>`	The menu the command appears in. If this tag is not provided, the command appears in the **File > Scripts** menu. **Note:** Currently the only supported value for `<type>` is `automate.` This value puts the script in the **File > Automate** menu.
`<about>`	A string that appears in an About box, which the user can select from the **Help > About Plug-in** menu.
`<enableinfo>`	A boolean expression that indicates whether the command is enabled in the menu. See Enable Info Grammar. **Note:** If you provide this tag, the menu item is enabled if and only if there is at least one document open, and the boolean expression evaluates to true. If you always want the menu item enabled, do not use this tag.
`<eventid>`	A unique string that identifies the event. Using a UUID will ensure that your script wont share this identifier with another script.

Tag	Description
`<category>`	The category the command appears within in the menu. Used to group and order commands in the menu. Commands are placed in the menu alphabetically based on the string in `<category>`. If two commands use the same category, they are grouped together.
`<terminology>`	The terminology dictionary for the script to function with the Action Manager. See the Terminology Dictionary.

Basic JavaScript Resource Example

This example shows a very basic `<javascriptresource>`. With this resource, the script can be executed by selecting the command **Add a Document**, which appears in the **Automate** menu. This command is enabled in the menu, provided at least one document is already open. If the user requests information about the script from the **About Plug-in** menu, the string contained in the `<about>` tag is displayed in a dialog box.

```
/*
   <javascriptresource>
   <name>Add a Document</name>
   <type>automate</type>
   <about>A short string providing information about the script.</about>
   <enableinfo>true</enableinfo>
   </javascriptresource>
*/

app.documents.add();
```

Enable Info Grammar

The `<enableinfo>` tag provides a boolean expression that, when evaluated, indicates whether the command is enabled in the menu. You can use this expression to enable or disable the menu item based on various characteristics of the document. The Enable Info grammar is as follows:

```
<booleanExpression> :=   <conjunction> { "||" <conjunction> }

<conjunction> :=         <relation> {"&&" <relation> }

<relation> :=            <equality> {<relationOperator><equality>}

<equality> :=            <simpleExpression> {<equalityOperator><simpleExpression>}

<simpleExpression> :=    <term> {<addOperator><term>}

<term> :=                <factor> {<mulOperator><factor>}

<factor> :=              <integer> | <intrinsic> | <ident> |
                         "(" <booleanExpression ")" | "(" simpleExpression ")" |
                         "+" <factor> | "-" <factor> | "!" <factor>

<integer> :=             digit {digit}

<intrinsic> :=           <limitFunction> | <dimFunction> | <inFunction>

<limitFunction> :=       ( "min" | "max") "(" <simpleExpression> ","
                         <simpleExpression> { "," <simpleExpression> } ")"

<dimFunction> :=         "dim" "(" <simpleExpression> "," <simpleExpression> ")"
```

```
<ident> :=                  (alpha | "_") {alpha | digit | "_" }

<mulOperator> :=            "*" | "/"

<addOperator> :=            "+" | "-"

<equalityOperator> :=       "==" | "!="

<relationOperator> :=       "<" | "<=" | ">=" | ">"

<inFunction> :=             "in" "(" <simpleExpression> {"," <simpleExpression> } ")"
```

Operator precedence is shown in the following table. Operators are listed with the highest order of precedence at the top of the table

Operator	Description
`||`	Or
`&&`	And
`+ -`	Addition or subtraction
`* /`	Multiply or divide
`< <= >= >`	Less than, less than or equal, greater than or equal, greater than
`== !=`	Equals, or does not equal.

`(..) in() max() min()`
`unary + - !`
	Functions
	Unary operators: increment, decrement, not

The grammar provides variables and constants that you can use in the `<enableinfo>` expression. The following table provides a list of the constants that are available.

Constant Name	Description
`true`	Boolean true
`false`	Boolean false
`BitmapMode`	Bitmap mode.
`GrayScaleMode`	Grayscale mode, 8 bit depth.
`IndexedMode`	Indexed color mode.
`RGBMode`	RGB color mode.
`CMYKMode`	CMYK color mode.
`HSLMode`	HSL color mode.
`HSBMode`	HSB color mode
`MultiChannelMode`	Multichannel mode.
`DuotoneMode`	Duotone mode.
`LabMode`	Lab color mode.

Constant Name	Description
Gray16Mode	Grayscale mode, 16 bits per channel
RGB48Mode	RGB color mode, 16 bits per channel.
Lab48Mode	LAB mode, 16 bits per channel.
CMYK64Mode	CMYK mode, 16 bits per channel.
DeepMultichannelMode	Deep multichannel mode.
Duotone16Mode	Duotone mode, 16 bit depth.
RGB96Mode	RGB color mode, 32 bits per channel.
Gray32Mode	Grayscale mode, 32 bit depth.

The following table show the set of variables you can use in the <enableinfo> expression. The value of these variables is set based on the properties of the active document.

Variable Name	Description
PSHOP_ImageMode	Image mode of the active document.
PSHOP_ImageDepth	Depth of the active document.
PSHOP_HasLayerMask	Boolean indicating presence of layer mask.
PSHOP_HasSelectionMask	Boolean indicating presence of selection mask.
PSHOP_HasTransparencyMask	Boolean indicating presence of transparency mask.
PSHOP_NumTargetChannels	Number of target channels.
PSHOP_NumTrueChannels	Numer of image channels.
PSHOP_IsAdjustorSheet	Boolean
PSHOP_IsTargetComposite	Boolean indicating whether channels are flattened.
PSHOP_IsTargetSection	Boolean.
PSHOP_IsTargetVisible	Boolean.
PSHOP_ImageWidth	Width of the image.
PSHOP_ImageHeight	Height of the image.
PSHOP_TargetProtectFlags.	

Undefined Values in Enable Info Evaluation

If any arithmetic or relation operation contains an operand whose value is undefined, or a variable that is undefined, the result of that evaluation is `false`.

Boolean values are treated as in C/C++, where non-zero values are `true`, and zero is `false`, with the exception that an undefined value is also `false`.

Using the “in” Function

The in function (see `<inFunction>`) returns `true` is the first parameter is equal to at least one of the subsequent parameters. A typical use might be to see if the image mode of the active document is one of a set of image modes. For example:

```
in(PSHOP_ImageMode, RGBMode, CMYKMode, LabMode)
```

Action Manager Automation

For your script to be able to record scripting parameters and be automated by them, it requires the addition of two basic mechanisms:

- A *terminology dictionary* that maps your script’s user interface to human readable text, providing text and type information for each parameter the script uses.
- Code to read parameter information when it comes from the Action Manager, rather than from the user-interface, and code to write parameter information to the Action Manger. This code uses the Action Manager classes ActionDescriptor, ActionList, and ActionReference.

See `Conditional Mode Change.jsx` for an example of a script that can record and be automated by scripting parameters.

Terminology Dictionary

The JavaScript resource provides a `<terminology>` tag that allows you to provide the terminology dictionary for your script. The first step in creating a terminology dictionary is to review your script’s user interface, and create human-readable strings for each element in your user interface.

For example, in the Conditional Mode Change command, the user interface requests a source mode and a target mode. Both source mode and target mode have several options. All of these elements of the user interface need to have entries in the terminology dictionary.

The terminology dictionary is created in a PDF dictionary format, with the following entries, and must have the following format in the `<javascriptresource>`:

```
<terminology><![CDATA[<<<
      /Version integer
      /Events <<event dictionary>>
      /Classes <<class dictionary>>
      /Enumerations <<enumeration dictionary>>
>>> ]]></terminology>
```

Note: The information in the terminology tag needs to be wrapped in a `CDATA` block so the xml parser will ignore “/” and other tags that appear in the terminology.

The defintions for events, classes and enumerations dictionaries are provided below.

The /Events dictionary contains an entry for each event:

```
/eventName [                    // Name used in string-based API
   (String event name)          // required
   /direct parameter type       // optional; if omitted, no parameter
      <<                        // optional parameter dictionary
      /parameterName [          // Name used in string-based API
         (String name)          // required
         /parameter type        // required
         ]
      ...                       // other parameters
      >>
   ]
```

The /Classes dictionary contains and entry for each class:

```
/className [                    // Name used in string-based API
   (ZString class name)         // required
      <<                        // property dictionary
      /propertyName [           // Name used in string-based API
         (String name)          // required
         /property type         // required
         ]
      ...                       // other properties
      >>
   ]
```

The /Enumerations dictionary contains an entry for each enumerated type:

```
/enumTypeName                   // Name used in string-based API
   <<
   /enumValue (String name)     // required
   ...
   >>
```

Value Type Defintions

For /parameter type and /property type definitons, you can use the Class and Enumeration type declarations you make in your own terminology dictionary, you can use declarations provided by Photoshop or you can use basic value types.

Basic Value Types

The basic value types are shown in the following table:

Name	Code	Description
typeInteger	'long'	int32
typeFloat	'doub'	IEEE 64 bit double
typeBoolean	'bool'	TRUE or FALSE.
typeText	'TEXT'	Block of any number of readable characters.
typeAlias	'alis'	Macintosh file system path.

Name	Code	Description
typePaths	'Pth '	Windows file system path.
typePlatformFilePath	'alis' or'Pth	typeAlias for Mac OS, typePath for Windows.

Predefined Class Types

Photoshop provides a number of predefined classes that are available for use in the terminology dictionary. A useful subset of those classes is shown in the table below. Use these classes when they are appropriate, but you can define new classes in the terminology resource, if necessary.

Name	Code	Description
classColor	'Clr '	Class for color classes.
classRGBColor	'RGBC'	keyRed, keyGreen, keyBlue
classCMYKColor	'CMYC'	keyCyan, keyMagenta, keyYellow, keyBlack.
classUnspecifiedColor	'UnsC'	Unspecified.
classGrayscale	'Grsc'	keyGray
classBookColor	'BkCl '	Book color
classLabColor	'LbCl'	keyLuminance, keyA, keyB.

Uniqueness Rules for Terminology Entries

Generally, the names for terminology entries must be unique within a particular category and scope. It is best to not make names unique unnecessarily; generic terms are prefereable, and if a name already exists for something, go ahead and use it. Case matters in considering uniqueness of terminology entries.

The uniqueness rules for terminology entries are:

- All event names must be different from all other event names.
- All class names must be different from all other names.
- All enumeration type names must be different from all other enumeration type names.
- All keys must be different from all other keys used in the same class or event.
- All enumeration values must be different from all other enumeration values in the same enumeration type.
- A class, event, enumeration type, key, and enumeration value can all have the same name.

Terminology Definition Example

This example demonstrates the terminology definition for a new event; the example uses ZStrings. The event is called newAnnot, and it takes three parameters:

- annotType, an enumeration (annotType)
- at, a class (point), and
- size, a class (annotSizeClass).

The annotSizeClass has two properties: width, and height, both of type floatType. The enumeration annotType has three values: annotUnknown, annotText, and annotSound.

```
<terminology><![CDATA[<<<
   /Version 1
   /Events
      /newAnnot [(New Annotation) <<
         /annotType [(Type) /annotType]
         /at  [(At) /Point]
         /size  [(Size) /annotSizeClass] >>]
   /Classes
      /annotSizeClass [(Size) <<
         /width [(Width) /floatType]
         /height [(Height) /floatType]
         >>]
   /Enumerations
      /annotType <<
         /annotUnknown (Unknown)
         /annotText (Text)
         /annotSound (Sound)
   >>> ]]></terminology>
```

Appendix A: Event ID Codes

The following table lists events and their four-character ID codes or string identifiers for use with the `notifier` object.

Note: Do not include single quotes (') with four-character IDs in your code. The single quotes are used in this table to illustrate the placement of required spaces in codes that do not contain four letters. However, string identifiers, which are longer than four characters, require double quotes in the code.

Tip: If you can't find the event you want to use for notification in this table, you can use ScriptListener to determine the event ID code. See the ScriptListener documentation in the Action Manager chapter of the *Photoshop CS3 Scripting Guide*.

Event	4-char ID or String
`3DTransform`	`'TdT '`
`Average`	`'Avrg'`
`ApplyStyle`	`'ASty'`
`Assert`	`'Asrt'`
`AccentedEdges`	`'AccE'`
`Add`	`'Add '`
`AddNoise`	`'AdNs'`
`AddTo`	`'AddT'`
`Align`	`'Algn'`
`All`	`'All '`
`AngledStrokes`	`'AngS'`
`ApplyImage`	`'AppI'`
`BasRelief`	`'BsRl'`
`Batch`	`'Btch'`
`BatchFromDroplet`	`'BtcF'`
`Blur`	`'Blr '`
`BlurMore`	`'BlrM'`
`Border`	`'Brdr'`
`Brightness`	`'BrgC'`
`CanvasSize`	`'CnvS'`
`ChalkCharcoal`	`'ChlC'`
`ChannelMixer`	`'ChnM'`

Event	4-char ID or String
Charcoal	'Chrc'
Chrome	'Chrm'
Clear	'Cler'
Close	'Cls '
Clouds	'Clds'
ColorBalance	'ClrB'
ColorHalftone	'ClrH'
ColorRange	'ClrR'
ColoredPencil	'ClrP'
ContactSheet	"0B71D221-F8CE-11d2-B21B-0008C75B322C"
ConteCrayon	'CntC'
Contract	'Cntc'
ConvertMode	'CnvM'
Copy	'copy'
CopyEffects	'CpFX'
CopyMerged	'CpyM'
CopyToLayer	'CpTL'
Craquelure	'Crql'
CreateDroplet	'CrtD'
Crop	'Crop'
Crosshatch	'Crsh'
Crystallize	'Crst'
Curves	'Crvs'
Custom	'Cstm'
Cut	'cut '
CutToLayer	'CtTL'
Cutout	'Ct '
DarkStrokes	'DrkS'
DeInterlace	'Dntr'
DefinePattern	'DfnP'
Defringe	'Dfrg'
Delete	'Dlt '
Desaturate	'Dstt'

Event	4-char ID or String
Deselect	'Dslc'
Despeckle	'Dspc'
DifferenceClouds	'DrfC'
Diffuse	'Dfs '
DiffuseGlow	'DfsG'
DisableLayerFX	'dlfx'
Displace	'Dspl'
Distribute	'Dstr'
Draw	'Draw'
DryBrush	'DryB'
Duplicate	'Dplc'
DustAndScratches	'DstS'
Emboss	'Embs'
Equalize	'Eqlz'
Exchange	'Exch'
Expand	'Expn'
Export	'Expr'
Jumpto	'Jpto'
ExportTransparentImage	"02879e00-cb66-11d1-bc43-0060b0a13dc4"
Extrude	'Extr'
Facet	'Fct '
Fade	'Fade'
Feather	'Fthr'
Fibers	'Fbrs'
Fill	'Fl '
FilmGrain	'FlmG'
Filter	'Fltr'
FindEdges	'FndE'
FitImage	"3caa3434-cb67-11d1-bc43-0060b0a13dc4"
FlattenImage	'FltI'
Flip	'Flip'
Fragment	'Frgm'
Fresco	'Frsc'

Event	4-char ID or String
GaussianBlur	'GsnB'
Get	'getd'
Glass	'Gls '
GlowingEdges	'GlwE'
Gradient	'Grdn'
GradientMap	'GrMp'
Grain	'Grn '
GraphicPen	'GraP'
Group	'GrpL'
Grow	'Grow'
HalftoneScreen	'HlfS'
Hide	'Hd '
HighPass	'HghP'
HSBHSL	'HsbP'
HueSaturation	'HStr'
ImageSize	'ImgS'
Import	'Impr'
InkOutlines	'InkO'
Intersect	'Intr'
IntersectWith	'IntW'
Inverse	'Invs'
Invert	'Invr'
LensFlare	'LnsF'
Levels	'Lvls'
LightingEffects	'LghE'
Link	'Lnk '
Make	'Mk '
Maximum	'Mxm '
Median	'Mdn '
MergeLayers	'Mrg2'
MergeLayersOld	'MrgL'
MergeSpotChannel	'MSpt'
MergeVisible	'MrgV'

Event	4-char ID or String
Mezzotint	'Mztn'
Minimum	'Mnm '
ModeChange	"8cba8cd6-cb66-11d1-bc43-0060b0a13dc4"
Mosaic	'Msc '
Mosaic_PLUGIN	'MscT'
MotionBlur	'MtnB'
Move	'move'
NTSCColors	'NTSC'
NeonGlow	'NGlw'
Next	'Nxt '
NotePaper	'NtPr'
Notify	'Ntfy'
Null	typeNull
OceanRipple	'OcnR'
Offset	'Ofst'
Open	'Opn '
Paint	'Pnt '
PaintDaubs	'PntD'
PaletteKnife	'PltK'
Paste	'past'
PasteEffects	'PaFX'
PasteInto	'PstI'
PasteOutside	'PstO'
Patchwork	'Ptch'
Photocopy	'Phtc'
PicturePackage	"4C1ABF40-DD82-11d2-B20F-0008C75B322C"
Pinch	'Pnch'
Place	'Plc '
Plaster	'Plst'
PlasticWrap	'PlsW'
Play	'Ply '
Pointillize	'Pntl'
Polar	'Plr '

Event	4-char ID or String
PosterEdges	'PstE'
Posterize	'Pstr'
Previous	'Prvs'
Print	'Prnt'
ProfileToProfile	'PrfT'
Purge	'Prge'
Quit	'quit'
RadialBlur	'RdlB'
Rasterize	'Rstr'
RasterizeTypeSheet	'RstT'
RemoveBlackMatte	'RmvB'
RemoveLayerMask	'RmvL'
RemoveWhiteMatte	'RmvW'
Rename	'Rnm '
ReplaceColor	'RplC'
Reset	'Rset'
ResizeImage	"1333cf0c-cb67-11d1-bc43-0060b0a13dc4"
Reticulation	'Rtcl'
Revert	'Rvrt'
Ripple	'Rple'
Rotate	'Rtte'
RoughPastels	'RghP'
Save	'save'
Select	'slct'
SelectiveColor	'SlcC'
Set	'setd'
SharpenEdges	'ShrE'
Sharpen	'Shrp'
SharpenMore	'ShrM'
Shear	'Shr '
Show	'Shw '
Similar	'Smlr'
SmartBlur	'SmrB'

Event	4-char ID or String
Smooth	'Smth'
SmudgeStick	'SmdS'
Solarize	'Slrz'
Spatter	'Spt '
Spherize	'Sphr'
SplitChannels	'SplC'
Sponge	'Spng'
SprayedStrokes	'SprS'
StainedGlass	'StnG'
Stamp	'Stmp'
Stop	'Stop'
Stroke	'Strk'
Subtract	'Sbtr'
SubtractFrom	'SbtF'
Sumie	'Smie'
TakeMergedSnapshot	'TkMr'
TakeSnapshot	'TkSn'
TextureFill	'TxtF'
Texturizer	'Txtz'
Threshold	'Thrs'
Tiles	'Tls '
TornEdges	'TrnE'
TraceContour	'TrcC'
Transform	'Trnf'
Trap	'Trap'
Twirl	'Twrl'
Underpainting	'Undr'
Undo	'undo'
Ungroup	'Ungr'
Unlink	'Unlk'
UnsharpMask	'UnsM'
Variations	'Vrtn'
Wait	'Wait'

Event	4-char ID or String
`WaterPaper`	`'WtrP'`
`Watercolor`	`'Wtrc'`
`Wave`	`'Wave'`
`Wind`	`'Wnd '`
`ZigZag`	`'ZgZg'`
`BackLight`	`'BacL'`
`FillFlash`	`'FilE'`
`ColorCast`	`'ColE'`

Index