JavaScript™ for Acrobat® API Reference

Adobe® Acrobat® SDK

April 2007

Version 8.1

© 2007 Adobe Systems Incorporated. All rights reserved.

Adobe® Acrobat® SDK 8.1 JavaScript for Acrobat API Reference for Microsoft® Windows® and Mac OS®.

Edition 2.0, April 2007

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.

Please remember that existing artwork or images that you may want to include in your project may be protected under copyright law. The unauthorized incorporation of such material into your new work could be a violation of the rights of the copyright owner. Please be sure to obtain any permission required from the copyright owner.

Any references to company names, company logos and user names in sample material or sample forms included in this documentation and/or software are for demonstration purposes only and are not intended to refer to any actual organization or persons.

Adobe, the Adobe logo, Acrobat, Distiller, FrameMaker, LiveCycle, PostScript and Reader are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Apple and Mac OS are trademarks of Apple Computer, Inc., registered in the United States and other countries.

JavaScript is a trademark or registered trademark of Sun Microsystems, Inc. in the United States and other countries.

Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

All other trademarks are the property of their respective owners.

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

Notice to U.S. Government End Users. The Software and Documentation are “Commercial Items,” as that term is defined at 48 C.F.R. §2.101, consisting of “Commercial Computer Software” and “Commercial Computer Software Documentation,” as such terms are used in 48 C.F.R. §12.212 or 48 C.F.R. §227.7202, as applicable. Consistent with 48 C.F.R. §12.212 or 48 C.F.R. §§227.7202-1 through 227.7202-4, as applicable, the Commercial Computer Software and Commercial Computer Software Documentation are being licensed to U.S. Government end users (a) only as Commercial Items and (b) with only those rights as are granted to all other end users pursuant to the terms and conditions herein. Unpublished-rights reserved under the copyright laws of the United States. Adobe Systems Incorporated, 345 Park Avenue, San Jose, CA 95110-2704, USA. For U.S. Government End Users, Adobe agrees to comply with all applicable equal opportunity laws including, if appropriate, the provisions of Executive Order 11246, as amended, Section 402 of the Vietnam Era Veterans Readjustment Assistance Act of 1974 (38 USC 4212), and Section 503 of the Rehabilitation Act of 1973, as amended, and the regulations at 41 CFR Parts 60-1 through 60-60, 60-250, and 60-741. The affirmative action clause and regulations contained in the preceding sentence shall be incorporated by reference.

Contents

Preface 28
What's in this guide? 28
Who should read this guide? 28
Related documentation 28

1 Introduction 30
Syntax 30
Paths 31
Safe path 31
Privileged context 31
Privileged versus non-privileged context 32
User preferences 32
Quick bars 33
Domain names in code samples 34

2 JavaScript API 35
ADBC 36
ADBC properties 36
SQL types 36
JavaScript types 37
ADBC methods 38
getDataSourceList 38
newConnection 38
Alerter 40
Alerter methods 40
dispatch 40
AlternatePresentation 43
AlternatePresentation properties 43
active 43
type 43
AlternatePresentation methods 44
start 44
stop 44
Annotation 45
Annotation types 45
Annotation properties 47
alignment 47
AP 48
arrowBegin 49
arrowEnd 50
attachIcon 50
author 51
borderEffectIntensity 51
borderEffectStyle 52
callout 52
caretSymbol 52

1 Introduction

JavaScript is the cross-platform scripting language of the Adobe Acrobat family of products that includes Acrobat Professional, Acrobat Standard, and Adobe Reader. Through JavaScript extensions, the viewer application and its plug-ins expose much of their functionality to document authors, form designers, and plug-in developers.

This functionality includes the following features, among others:

- Processing forms within the document
- Batch processing collections of PDF documents
- Developing and maintaining online collaboration schemes
- Communicating with local databases
- Controlling multimedia events

In addition to being available in Acrobat and Adobe Reader, the objects, properties, and methods for the Acrobat extensions for JavaScript can also be accessed through Microsoft Visual Basic to automate the processing of PDF documents. See the *Interapplication Communication API Reference* for details.

Syntax

Some JavaScript objects are *static* objects that can be used as is and must be spelled as indicated. For example, the `app` object represents the JavaScript application. There is only one such object and it must be spelled `app` (case-sensitive).

Other objects are dynamic objects that can be assigned to a variable. For example, a Doc object may be obtained and assigned to a variable:

```
var myDoc = app.newDoc();
```

In this example, `myDoc` can access all methods and properties of the Doc object. For example:

```
myDoc.closeDoc();
```

Method arguments

Many of the JavaScript methods provided by Acrobat accept either a list of arguments, as is customary in JavaScript, or a single object argument with properties that contain the arguments. For example, these two calls are equivalent:

```
app.alert( "Acrobat Multimedia", 3);

app.alert({ cMsg: "Acrobat Multimedia", nIcon: 3});
```

Note: The JavaScript methods defined in support of multimedia do not accept these two argument formats interchangeably. Use the exact argument format described for each method.

Parameter help

When using Acrobat Professional, if you give an Acrobat method an argument of `acrohelp` and execute that method in the JavaScript Debugger console (or any internal JavaScript editor), the method returns a list of its own arguments.

For example, enter the following code in the console window.

```
app.response(acrohelp)
```

While the cursor is still on the line just entered, press either Ctrl + Enter or the Enter key on the numeric pad. The console displays the following message.

```
HelpError: Help.
app.response:1:Console undefined:Exec
====> [cQuestion: string]
====> [cTitle: string]
====> [cDefault: string]
====> [bPassword: boolean]
====> [cLabel: string]
```

Parameters listed in square brackets indicate optional parameters.

Note: Parameter help is not implemented for every JavaScript method. For example, it is not implemented for methods defined in the App JavaScript folder.

Paths

Several methods take *device-independent paths* as arguments. See the *PDF Reference*, version 1.7, for details about the device-independent path format.

Safe path

Acrobat 6.0 introduced the concept of a *safe path* for JavaScript methods that write data to the local hard drive based on a path passed to it by one of its parameters.

A path cannot point to a system critical folder, for example a root, windows or system directory. A path is also subject to other unspecified tests.

For many methods, the file name must have an extension appropriate to the type of data that is to be saved. Some methods may have a no-overwrite restriction. These additional restrictions are noted in the documentation.

Generally, when a path is judged to be not safe, a `NotAllowedError` exception is thrown (see `Error` object) and the method fails.

Privileged context

A context in which you have the right to do something that is normally restricted. Such a right (or privilege) could be granted by executing a method in a specific way (through the console or batch process), by some PDF property, or because the document was signed by someone you trust. For example, trusting a document certifier's certificate for executing JavaScript creates a privileged context which enables the JavaScript to run where it otherwise would not.

Privileged versus non-privileged context

Some JavaScript methods, marked in this book by Ⓢ in the third column of the quick bar, have security restrictions. These methods can be executed only in a *privileged context*, which includes console, batch and application initialization events. All other events (for example, page open and mouse-up events) are considered *non-privileged*.

The description of each security-restricted method indicates the events during which the method can be executed.

Beginning with Acrobat 6.0, security-restricted methods can execute without restrictions if the document certifier's certificate is trusted for running embedded high privilege JavaScript.

In Acrobat versions earlier than 7.0, menu events were considered privileged contexts. Beginning with Acrobat 7.0, execution of JavaScript through a menu event is no longer privileged. You can execute security-restricted methods through menu events in one of the following ways:

- By opening the JavaScript category of the Acrobat preferences and checking the item named "Enable Menu Items JavaScript Execution Privileges".
- By executing a specific method through a *trusted function* (introduced in Acrobat 7.0). Trusted functions allow privileged code—code that normally requires a privileged context to execute—to execute in a non-privileged context. For details and examples, see `app.trustedFunction`.

User preferences

There are many references in this document to the Acrobat user preferences. The preferences dialog box is accessed through the following menu commands, depending on platform:

Microsoft® Windows®: Edit > Preferences

Mac OS: Acrobat > Preferences

The preferences dialog box contains several categories that have relevant commands, including Forms, General, and JavaScript.

The following methods, if run from a document-level script, no longer affect the user preferences:

- `app.fs.defaultTransition, app.fsTransition`
- `app.fs.useTimer, app.fsUseTimer`
- `app.fs.usePageTiming, app.fsUsePageTiming`
- `app.fs.loop, app.fsLoop`
- `app.fs.escapeExits, app.fsEscape`
- `app.fsClick, app.fs.clickAdvances`
- `app.fsTimeDelay, app.fs.timeDelay`
- `app.fsColor, app.fs.backgroundColor`
- `app.fsCursor, app.fs.cursor`
- `app.openInPlace`

These methods still affect user preferences if run from an application-level script.

Also note that `app.fs.escapeExits` and `app.fsEscape` can now only be set to `false` when running in a privileged context.

Quick bars

At the beginning of most property and method descriptions, a small table or *quick bar* provides a summary of the item's availability and usage recommendations.

The quick bar shown here has descriptive column headings that are not shown in the reference.

Version or Deprecated	Save and Preferences	Security	Availability
6.0	Ⓓ	Ⓢ	Ⓒ

The following tables show the symbols that can appear in each column and their meanings.

Column 1: Version or deprecated

#.# A number indicates the version of the software in which a property or method became available. If the number is specified, the property or method is available only in versions of the Acrobat software greater than or equal to that number.

For Acrobat 8.0, there are some compatibility issues with older versions. Before accessing the property or method, the script should check that the forms version is greater than or equal to that number to ensure backward compatibility. For example:

```
if (typeof app.formsVersion != "undefined" && app.formsVersion >= 8.0)
{
   // Perform version specific operations.
}
```

If the first column is blank, no compatibility checking is necessary.

Ⓧ The property or method is deprecated.

Column 2: Save and Preferences

Ⓓ Writing to this property or method dirties (modifies) the PDF document. If the document is subsequently saved, the effects of this method are saved as well. (In Adobe Reader, the document requires specific rights to be saved.)

Ⓟ Even though this property does not change the document, it can permanently change a user's application preferences.

Column 3: Security

Ⓢ	For security reasons, this property or method may be available only during certain events. These events include batch processing, application start, or execution within the console. (See the `event` object for details of the Acrobat events.) Beginning with Acrobat 7.0, to execute a security-restricted method through a menu event, one of the following must be true: • The JavaScript user preferences item Enable Menu Items JavaScript Execution Privileges is checked. • The method is executed through a trusted function. For details and examples, see the `app.trustedFunction` method. See "Privileged versus non-privileged context" on page 32 for more information. **Note:** (Acrobat 6.0 or later) Methods marked with Ⓢ will execute without restriction in a certified document provided the certifier's certificate is trusted for running embedded high privilege JavaScript and other limitations in the quick bar fields are met.

Column 4: Availability

	If the column is blank, the property or method is allowed in Adobe Reader, Acrobat Professional or Acrobat Standard.
Ⓧ	The property or method is not allowed in Adobe Reader but is available in Acrobat Professional and Acrobat Standard.
Ⓕ Ⓒ Ⓢ Ⓓ Ⓖ	The property or method is allowed in Acrobat Professional and Acrobat Standard. It can be accessed in Adobe Reader (version 5.1 or later) depending on additional usage rights that have been applied to the document: • F — Requires forms rights • C — Requires the right to manipulate comments • S — Requires the document save right • D — Requires file attachment rights • G — Requires digital signature rights
Ⓟ	The property or method is available only in Acrobat Professional

Domain names in code samples

Throughout this document there are numerous code samples that use URLs. Such examples use the domain names example.com, example.net, and example.org, which are reserved for the purpose of illustration. Some examples use IP addresses, these addresses come from the range 172.16.0.0 through 172.31.255.255, which are reserved for private networks.