



# JavaScript™ for Acrobat® 3D Annotations API Reference

February 2011

**Adobe® Acrobat® SDK**

Version 10.0

© 2011 Adobe Systems Incorporated. All rights reserved.

Adobe® Acrobat® X SDK JavaScript™ for Acrobat 3D Annotations API Reference for Microsoft® Windows® and Mac OS®

Edition 4.0, February 2011

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, and Reader are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

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

Mac OS is a trademark 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/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

---

<b>Preface .....</b>	<b>8</b>
What's in this guide? .....	8
Who should read this guide? .....	8
Related documentation .....	8
<b>1 Introduction .....</b>	<b>9</b>
Object overview .....	11
Basic objects .....	11
Scene object .....	11
Canvas object .....	11
Runtime object .....	12
Console object .....	12
Resource objects .....	12
Event handlers .....	12
CamaraEvent.....	12
KeyEvent.....	13
MenuEvent .....	13
MouseEvent .....	13
RenderEvent.....	13
ScrollWheelEvent .....	14
SelectionEvent .....	14
TimeEvent.....	14
ToolEvent.....	14
<b>2 JavaScript Objects for Acrobat 3D .....</b>	<b>15</b>
Animation .....	16
Background .....	17
getColor.....	17
getImage .....	17
setColor.....	17
setImage.....	18
Bone .....	19
BoundingBox .....	20
Camera .....	21
getScreenFromPosition .....	22
getDirectionFromScreen.....	23
CamaraEvent.....	24
CameraEventHandler.....	25
CameraEventHandler .....	25
onEvent.....	25
Canvas .....	26
getCamera .....	26
setCamera .....	26
ClippingPlane .....	27
remove.....	27
Color.....	28
Color.....	28

Color.....	28
set .....	28
set .....	29
set3 .....	29
Console .....	31
print.....	31
println .....	31
Dummy .....	32
FlashEvent.....	33
FlashEventHandler .....	34
onEvent.....	34
FlashEventHandler.....	34
FlashMovie .....	35
FlashMovie.....	37
call.....	37
getVariable .....	38
gotoFrame .....	38
isPlaying .....	39
pan.....	39
play .....	39
rewind .....	39
setVariable .....	40
setZoomRect.....	40
stop .....	41
zoom .....	41
HitInfo.....	42
Host .....	42
Image .....	43
Image.....	43
KeyEvent.....	44
EventHandler .....	46
EventHandler .....	46
onEvent.....	46
Light .....	47
Material .....	49
attachFlashMovie.....	50
Matrix4x4.....	51
Matrix4x4 .....	51
Matrix4x4 .....	51
invertInPlace .....	52
isEqual .....	52
multiply.....	52
multiplyInPlace .....	52
rotateWithQuaternion.....	53
rotateWithQuaternionInPlace .....	53
rotateAboutLine .....	53
rotateAboutLineInPlace.....	54
rotateAboutX.....	54
rotateAboutXInPlace .....	55
rotateAboutVector .....	55
rotateAboutVectorInPlace .....	55

rotateAboutY .....	56
rotateAboutYInPlace .....	56
rotateAboutZ .....	56
rotateAboutZInPlace .....	57
scale .....	57
scaleInPlace .....	58
set .....	58
set .....	58
set .....	59
setIdentity .....	59
setView .....	59
transformDirection .....	60
transformPosition .....	60
translate .....	60
translateInPlace .....	61
transposeInPlace .....	61
MenuEvent .....	62
MenuEventHandler .....	63
MenuEventHandler .....	63
onEvent .....	63
Mesh .....	64
computeBoundingBox .....	64
MouseEvent .....	65
MouseEventHandler .....	67
MouseEventHandler .....	68
onEvent .....	68
Node .....	69
detachFromCurrentAnimation .....	70
Procedural .....	71
Quaternion .....	72
Quaternion .....	72
Quaternion .....	72
Quaternion .....	72
interpolate .....	73
interpolateInPlace .....	73
normalize .....	73
RenderEvent .....	75
RenderEventHandler .....	76
RenderEventHandler .....	76
onEvent .....	76
RenderOptions .....	77
Resource .....	79
Resource .....	79
Runtime .....	80
addCustomMenuItem .....	83
addCustomToolButton .....	83
addEventHandler .....	84
disableTool .....	84
enableTool .....	84
getEventHandler .....	85
getRendererName .....	85

getView .....	85
getView .....	86
pause .....	86
play .....	86
refresh .....	87
removeEventHandler .....	87
removeCustomMenuItem .....	87
removeCustomToolButton .....	88
setCurrentTool .....	88
setCustomMenuItemChecked .....	88
setView .....	89
setView .....	89
Scene .....	91
activateAnimation .....	97
addFlashForeground .....	97
addModel .....	98
createClippingPlane .....	98
createLight .....	98
createSquareMesh .....	98
computeBoundingBox .....	99
update .....	99
SceneObject .....	100
SceneObjectList .....	101
getByGUID .....	101
getByID .....	101
getByIndex .....	101
getByName .....	102
removeAll .....	102
removeByIndex .....	102
removeItem .....	103
ScrollWheelEvent .....	104
ScrollWheelEventHandler .....	105
ScrollWheelEventHandler .....	105
onEvent .....	105
SelectionEvent .....	106
SelectionEventHandler .....	107
SelectionEventHandler .....	107
onEvent .....	107
StateEvent .....	108
StateEventHandler .....	109
onEvent .....	109
Syntax .....	109
StateEventHandler .....	109
Syntax .....	109
Texture .....	110
getImage .....	110
setImage .....	111
TimeEvent .....	112
TimeEventHandler .....	113
TimeEventHandler .....	113
onEvent .....	113

ToolEvent .....	114
ToolEventHandler .....	115
ToolEventHandler .....	115
onEvent.....	115
Vector3 .....	116
Vector3.....	116
Vector3.....	116
add.....	117
addInPlace .....	117
addScaled .....	117
addScaledInPlace .....	118
blend.....	118
blendInPlace .....	118
cross .....	119
dot .....	119
normalize .....	119
scale .....	120
scaleInPlace .....	120
set .....	120
set .....	121
set3 .....	121
subtract.....	122
subtractInPlace .....	122
View .....	123
<b>3 New Features and Changes.....</b>	<b>124</b>
Acrobat X changes .....	124
Acrobat 9.0 changes .....	124
Acrobat 8.1 changes .....	125
Acrobat 8.0 changes .....	126
<b>Index .....</b>	<b>127</b>

---

# Preface

---

The JavaScript™ API lets you manipulate 3D annotations within Adobe® PDF documents.

## What's in this guide?

This document provides a brief overview of the API followed by a description of the objects.

## Who should read this guide?

This guide is for developers who want to enhance the 3D experience of the user beyond the default behaviors. Using the JavaScript API for 3D annotations, you can specify the render modes and 3D matrix transformations of any of the individual meshes; set camera position, target, and field of view; detect mouse and keyboard events; control animations; and many more behaviors.

## Related documentation

This document refers to the following sources for additional information about 3D annotations, JavaScript, and related technologies. The Adobe Acrobat® documentation is available through the [Acrobat Developer Center](#).

Document	Description
<i>Developing Acrobat Applications Using JavaScript</i>	Using JavaScript to develop and enhance standard workflows in Acrobat and Adobe Reader®.
<i>JavaScript for Acrobat API Reference</i>	Detailed descriptions of JavaScript APIs for developing and enhancing workflows in Acrobat and Adobe Reader.
<i>PDF Reference</i>	A detailed description of the PDF file format.



## Introduction

To create 3D annotations and to attach scripts to them using this API, you need Adobe® Acrobat® Professional. Scripts attached to 3D annotations can run on Acrobat Pro Extended, Acrobat Pro, Acrobat Standard, and Adobe Reader® for Windows® and Mac OS® platforms. Unless otherwise noted, all JavaScript objects, properties, and methods have support starting in version 7.0.

The 3D JavaScript engine, which is distinct from the JavaScript engine for Acrobat, can be accessed in one of two ways. The primary way is by attaching a default script to the 3D annotation. This can be accomplished while placing a 3D annotation using the 3D Tool or on an existing 3D annotation by accessing its properties dialog box using the Select Object tool. This script will be run directly by the 3D JavaScript engine.

In addition, Acrobat provides a mechanism to directly access the entire 3D JavaScript engine API from within the Acrobat scripting engine by means of the JavaScript `Annot3D.context3D` property. For more information about JavaScript for Acrobat and its `Annot3D` object, see the [JavaScript for Acrobat API Reference](#) and [Developing Acrobat Applications Using JavaScript](#).

The following example illustrates how to access the 3D JavaScript engine. In this example, a button (or link) contains JavaScript code that rotates the U3D object named "Axes".

```
// Get index of page containing the Annot3D object (count starts at 0).
pageIndex = this.pageNum;

// Index of the Annot3D (count starts at 0).
annotIndex = 0;

// Get a reference to the Annot3D script context.
c3d = this.getAnnots3D( pageIndex )[ annotIndex ].context3D;

// Get a reference to the node in the scene named "Axes".
axes = c3d.scene.nodes.getByName( "Axes" );

// Rotate the object about the X-Axis PI/6 radians (30 degrees).
axes.transform.rotateAboutXInPlace( Math.PI / 6 );
```

More extensive actions can be executed by having a button or link get the `SceneContext3d` object and call a function defined in the default script of the 3D annotation, as in the following example.

```
// Get the Annot3D script context of the targeted annotation.
context3D = getAnnots3D(0) [0].context3D;

// Call the JavaScript function setRenderMode() defined in the default
// script of the referenced 3D annotation.
context3D.setRenderMode("transparent");
```

The default script of the 3D annotation makes the definition.

```
function setRenderMode( renderModeName ) {
    for (var i=0; i < scene.meshes.count; i++) {
```

```
        scene.meshes.getByIndex(i).renderMode = renderModeName;  
    }  
}
```

## Object overview

This section provides an overview of the objects in the 3D JavaScript API.

### Basic objects

There are several basic objects, such as `Color`, `Matrix4x4`, and `Vector3`, that are used to create general-purpose objects. The basic objects are used throughout the API and are only meaningful when attached to objects such as `Scene` or `Runtime`. For example, you could create a `Color` object and use it to set the Background color of a `Canvas`.

#### Vector3 Examples

```
v1 = new Vector3( 1.2, 3, 4.5 );  
v2 = new Vector3( 5, 8, 13 );  
v3 = new Vector3();
```

#### Matrix4x4 Examples

```
m1 = new Matrix4x4().rotateAboutX(Math.PI/1.5).rotateAboutY(Math.PI/3);  
m2 = new Matrix4x4().rotateAboutZ(Math.PI/4).translate(new Vector3(0,5,0));  
m3 = new Matrix4x4(m1);
```

#### Color Examples

```
c1 = new Color( 0.6, 0.8, 1.0 ); // light blue  
c2 = new Color( 0.5, 0.5, 0.5 ); // middle grey  
c3 = new Color(); //black  
  
// A function to blend two Colors  
Color.prototype.blend = function( color, amount )  
{  
    red      = ( this.r * ( 1 - amount ) ) + ( color.r * amount );  
    green    = ( this.g * ( 1 - amount ) ) + ( color.g * amount );  
    blue     = ( this.b * ( 1 - amount ) ) + ( color.b * amount );  
    return( new Color( red, green, blue ) );  
}  
c4 = c1.blend( c2, 0.25 );
```

### Scene object

The `Scene` is an object that contains all of the 3D-related content. It can be accessed using the global variable `scene`, which is a reference to the main `Scene` object. Most of the contents of the `Scene` are structured into a hierarchy of `Node` objects, and maintains lists of all these objects in the form of a `SceneObjectList`.

For more information, see [Scene](#).

### Canvas object

Represents a rectangular region into which a `Scene` is rendered from a particular viewpoint.

For more information, see [Canvas](#).

## Runtime object

The `Runtime` object is used to represent the instance of the playback engine. It manages all event processing and places where the graphic and textual content is rendered. It is accessed via the global variable `runtime`, which is a reference to the main `Runtime` object.

For more information, see [Runtime](#).

## Console object

The Console is the Acrobat text output area. It is helpful in debugging scripts.

## Resource objects

Some objects, such as `Image`, are driven by content that is streamed from a file or over a network. To create an `Image`, load a .png, .jpg, or .gif file as a `Resource`, which you may subsequently use to create a new `Image` object, as shown in the following example:

```
faceRes = new Resource("pdf://picture.jpg");  
faceImage = new Image( faceRes );  
aMaterial = scene.meshes.getByIndex(0).material;  
aMaterial.diffuseTexture.setImage( faceImage );
```

For more information, see [Resource](#) and [Image](#).

## Event handlers

There are several types of event handlers:

- [CameraEventHandler](#)
- [KeyEventHandler](#)
- [MouseEventHandler](#)
- [MenuEventHandler](#)
- [RenderEventHandler](#)
- [ScrollWheelEventHandler](#)
- [SelectionEventHandler](#)
- [TimeEventHandler](#)
- [ToolEventHandler](#)

Each one responds to a different type of event during simulation. They use a callback mechanism to run a function when an event occurs. The event is passed as an argument to the event handler's `onEvent` function so that it can be queried when the function runs. Event handlers are registered via the [addEventHandler](#) method, of the `Runtime` object.

## CamaraEvent

A `CamaraEvent` is created when a `View` is selected.

For more information, see [CamaraEvent](#).

## KeyEvent

A `KeyEvent` is created when a key is pressed or released while the 3D Canvas is in focus. The following example illustrates how to handle a key event:

```
myKeyHandler          = new KeyEventHandler();  
myKeyHandler.onEvent  = function( event )  
{  
    console.print( "Key pressed with code: " + event.characterCode );  
}  
runtime.addEventHandler( myKeyHandler );
```

For more information, see [KeyEvent](#).

## MenuEvent

A `MenuEvent` is created when a custom menu item is selected. To create a custom menu item on the context menu, invoke the `Runtime` object's `addCustomMenuItem` method, which allows a script to be attached to the item selection event.

For more information, see [MenuEvent](#).

## MouseEvent

A `MouseEvent` is created when the mouse is clicked on an active 3D Canvas or the cursor moves over an active 3D Canvas. The following syntax could be used to handle a mouse event:

```
myMouseHandler          = new MouseEventHandler();  
myMouseHandler.onMouseDown = true;  
myMouseHandler.target    = scene.meshes.getByIndex(0);  
myMouseHandler.onEvent   = function( event )  
{  
    console.print( "Mouse down at pixel " + event.mouseX );  
    console.print( ", " + event.mouseY );  
}  
runtime.addEventHandler( myMouseHandler );
```

For more information, see [MouseEvent](#).

## RenderEvent

A `RenderEvent` is created immediately before an instance of the `Canvas` is drawn. If there is a split view in Acrobat resulting in two visible 3D rendered areas, a unique `RenderEvent` will be called for each of them. This is necessary in the case of a camera-aligned image (sprite) in the 3D content that needs to be pixel-aligned. Since the pixel dimensions of the two areas are possibly different, there are two callbacks that pass the different dimensions. This makes it possible to modify the `Scene` in the appropriate manner before it is drawn.

For more information, see [RenderEvent](#).

## ScrollWheelEvent

A `ScrollWheelEvent` object is created when the mouse scroll wheel is activated over an active 3D Canvas object.

For more information, see [ScrollWheelEvent](#).

## SelectionEvent

A `SelectionEvent` object is created when an object is selected from an active 3D Canvas object or from a model tree. If the selection is made from a Canvas object, a `MouseEvent` is also created.

For more information, see [SelectionEvent](#).

## TimeEvent

A `TimeEvent` is created when the 3D annotation is enabled and simulation is active. The time and `deltaTime` properties are measured in terms of simulation time, not real time. `TimeEvent` objects are used to drive animation. If you need an accurate, real-time measurement, use the JavaScript `Date` object. The following syntax is used to handle a time event:

```
myTimeHandler          = new TimeEventHandler();  
myTimeHandler.onEvent  = function( event )  
{  
    console.print( "Current simulation time is:" + event.time );  
    console.print( " second(s)" );  
}  
runtime.addEventHandler( myTimeHandler );
```

For more information, see [TimeEvent](#).

## ToolEvent

A `ToolEvent` is created when a tool is clicked in the Acrobat 3D toolbar. The `Runtime` object's `addCustomToolButton` method allows you to add a custom tool to the toolbar which will also be generated, and allows a script to be attached to the tool selection event.

For more information, see [ToolEvent](#).

## 2

## JavaScript Objects for Acrobat 3D

---

This chapter describes the following 3D JavaScript objects:

<a href="#"><u>Animation</u></a>	<a href="#"><u>MouseEvent</u></a>
<a href="#"><u>Background</u></a>	<a href="#"><u>MouseEventHandler</u></a>
<a href="#"><u>Bone</u></a>	<a href="#"><u>Node</u></a>
<a href="#"><u>BoundingBox</u></a>	<a href="#"><u>Procedural</u></a>
<a href="#"><u>Camera</u></a>	<a href="#"><u>Quaternion</u></a>
<a href="#"><u>CameraEvent</u></a>	<a href="#"><u>RenderEvent</u></a>
<a href="#"><u>CameraEventHandler</u></a>	<a href="#"><u>RenderEventHandler</u></a>
<a href="#"><u>Canvas</u></a>	<a href="#"><u>RenderOptions</u></a>
<a href="#"><u>ClippingPlane</u></a>	<a href="#"><u>Resource</u></a>
<a href="#"><u>Color</u></a>	<a href="#"><u>Runtime</u></a>
<a href="#"><u>Console</u></a>	<a href="#"><u>Scene</u></a>
<a href="#"><u>Dummy</u></a>	<a href="#"><u>SceneObject</u></a>
<a href="#"><u>FlashEvent</u></a>	<a href="#"><u>SceneObjectList</u></a>
<a href="#"><u>FlashEventHandler</u></a>	<a href="#"><u>ScrollWheelEvent</u></a>
<a href="#"><u>FlashMovie</u></a>	<a href="#"><u>ScrollWheelEventHandler</u></a>
<a href="#"><u>HitInfo</u></a>	<a href="#"><u>SelectionEvent</u></a>
<a href="#"><u>Host</u></a>	<a href="#"><u>SelectionEventHandler</u></a>
<a href="#"><u>Image</u></a>	<a href="#"><u>StateEvent</u></a>
<a href="#"><u>KeyEvent</u></a>	<a href="#"><u>StateEventHandler</u></a>
<a href="#"><u>KeyEventHandler</u></a>	<a href="#"><u>Texture</u></a>
<a href="#"><u>Light</u></a>	<a href="#"><u>TimeEvent</u></a>
<a href="#"><u>Material</u></a>	<a href="#"><u>TimeEventHandler</u></a>
<a href="#"><u>Matrix4x4</u></a>	<a href="#"><u>ToolEvent</u></a>
<a href="#"><u>MenuEvent</u></a>	<a href="#"><u>ToolEventHandler</u></a>
<a href="#"><u>MenuEventHandler</u></a>	<a href="#"><u>Vector3</u></a>
<a href="#"><u>Mesh</u></a>	<a href="#"><u>View</u></a>

**Note:** A property labeled as R (read-only) is one whose value cannot be set. An object labeled as R (read-only) is one whose reference cannot be modified, though the object itself can be set and its properties may be modified. Unless otherwise indicated, all properties and objects labeled with R/W have read/write access.

## Animation

A type of [SceneObject](#), used to store keyframe animation sequences of `Node` objects in the Scene. In addition to the methods and properties below, it also contains the same methods and properties as a `SceneObject`.

### Properties

Property	Type	Access	Description
<code>currentTime</code>	number	R/W	The current time measured in seconds.
<code>endTime</code>	number	R	The end time of the sequence, measured in seconds.
<code>framesPerSecond</code>	number	R	The number of frames per second used to author the sequence.
<code>length</code>	number	R	The length of the <code>Animation</code> , measured in seconds.
<code>startTime</code>	number	R	The start time of the sequence, measured in seconds.



# Background

Represents the background of a `Canvas`. It can be used as a target of a `MouseEventHandler`. (See [Canvas](#) and [MouseEventHandler](#).)

## Properties

Property	Type	Access	Description
<code>image</code>	<code>Image</code>	R/W	Acrobat 7.0.7 The <code>Image</code> to be used by the <code>Background</code> .
<code>FlashMovie</code>	<code>FlashMovie</code>	R/W	Acrobat 9.0 The <code>FlashMovie</code> to be used by the <code>Background</code> . <code>FlashMovie</code> replaces any <code>Image</code> or <code>Color</code> currently being used by the <code>Background</code>

## getColor

Obtains the background `Color`.

## Syntax

```
getColor()
```

## Returns

A `Color` object representing the background color of the `Canvas`.

## getImage

Deprecated

Obtains the background `Image`.

## Syntax

```
getImage()
```

## Returns

An `Image` object representing the background image of the `Canvas`.

## setColor

Sets the background `Color`. If only one color is passed to this method, the background is a constant color. If two colors are passed to this method, the background is a linear gradient from top to bottom, with the first color argument representing the top color and the second representing the bottom color.

## Syntax

```
setColor(topColor, bottomColor)
```

## Parameters

topColor	A <code>Color</code> object representing the desired background color. If <code>bottomColor</code> is used, <code>topColor</code> represents the top background color used in a linear gradient.
bottomColor	(Optional) A <code>Color</code> object representing the bottom background color used in a linear gradient.

## Returns

undefined

## setImage

Deprecated

Sets the background Image.

## Syntax

```
setImage (image)
```

## Parameters

image	An <code>Image</code> object representing the desired background image.
-------	---

## Returns

undefined

## Bone

A type of `Node` used to modify the shape of a `Mesh`, and is usually moved over time to create animated characters. It contains the same methods and properties as a `Node`.

Related objects are [Node](#) and [Mesh](#).

# BoundingBox

Represents an axis-aligned bounding box.

## Properties

Property	Type	Access	Description
center	Vector3	R	Acrobat 7.0.7 The coordinates of the BoundingBox center.
max	Vector3	R	The coordinates of the BoundingBox corner with the greatest x, y, and z values.
min	Vector3	R	The coordinates of the BoundingBox corner with the smallest x, y, and z values.

## Camera

A `Node` that controls the projection from world space to screen space. In addition to the methods and properties below, it also contains the same methods and properties as a [Node](#).

### Properties

Property	Type	Access	Description
<code>absoluteBindingScale</code>	number	R/W	Acrobat 7.0 The absolute binding scale value for the camera.
<code>binding</code>	string	R/W	The view plane calculation type, which can take one of the following values: <ul style="list-style-type: none"><li>• "min"</li><li>• "max"</li><li>• "horizontal"</li><li>• "vertical"</li></ul>
<code>BINDING_HORIZONTAL</code>	string	R	Acrobat 7.0.7 A string constant for the binding value of "horizontal".
<code>BINDING_MAX</code>	string	R	Acrobat 7.0.7 A string constant for the binding value of "max".
<code>BINDING_MIN</code>	string	R	Acrobat 7.0.7 A string constant for the binding value of "min".
<code>BINDING_VERTICAL</code>	string	R	Acrobat 7.0.7 A string constant for the binding value of "vertical".
<code>far</code>	number	R/W	The distance from the <code>Camera</code> to the far clipping plane. A value of -1 for both <code>near</code> and <code>far</code> signifies to use auto clipping plane calculations.
<code>fov</code>	number	R/W	The size of the field of view for perspective <code>Camera</code> objects, measured in radians.
<code>near</code>	number	R/W	The distance from the <code>Camera</code> to the near clipping plane. A value of -1 for both <code>near</code> and <code>far</code> signifies to use auto clipping plane calculations.
<code>position</code>	Vector3	R	The position of the origin of the <code>Camera</code> in world space.
<code>positionLocal</code>	Vector3	R	The position of the origin of the <code>Camera</code> in local space.

Property	Type	Access	Description
projectionType	string	R/W	The type of projection, which can take one of the following values: <ul style="list-style-type: none"><li>• "perspective"</li><li>• "orthographic"</li></ul>
roll	number	R/W	The roll angle of the Camera, measured in radians.
target	Node	R	The current Node used as the Camera object's target.
targetPosition	Vector3	R	The position of the Camera object's target in world space.
targetPositionLocal	Vector3	R/W	The position of the Camera object's target in local space.
TYPE_ORTHOGRAPHIC	string	R	Acrobat 7.0.7 A string constant for the camera projection type of "orthographic".
TYPE_PERSPECTIVE	string	R	Acrobat 7.0.7 A string constant for the camera projection type of "perspective".
up	Vector3	R	The up direction in world space.
upLocal	Vector3	R	The up direction in local space.
useAbsoluteBinding	Boolean	R	Acrobat 7.0 Determines whether the camera uses absolute binding for its projection.
viewPlaneSize	number	R/W	The size of the view plane for orthographic Camera objects, measured in scene units.

## getScreenFromPosition

Obtains the screen coordinates of the provided 3D position.

### Syntax

```
getScreenFromPosition(position, canvasWidth, canvasHeight)
```

### Parameters

position	A Vector3 object representing the 3D position.
----------	--

canvasWidth	The width of the Canvas, measured in pixels.
canvasHeight	The height of the Canvas, measured in pixels.

## Returns

A `Vector3` object representing the screen coordinates, with `x` and `y` as pixel positions and `z` equal to zero.

See [Vector3](#) for more information on the return object.

## getDirectionFromScreen

Obtains the direction from the normalized coordinates

## Syntax

```
getDirectionFromScreen(x, y, canvasWidth, canvasHeight)
```

## Parameters

<code>x</code>	The x-coordinate, measured in pixels.
<code>y</code>	The y-coordinate, measured in pixels.
<code>canvasWidth</code>	The width of the Canvas, measured in pixels.
<code>canvasHeight</code>	The height of the Canvas, measured in pixels.

## Returns

A `Vector3` object representing the direction.

See [Vector3](#) for more information on the return object.

## CamaraEvent

Describes the format of the object that is passed as an argument to the `onEvent` method of the `CameraEventHandler` object.

### Properties

Property	Type	Access	Description
<code>binding</code>	string	R	The view plane calculation type, which can take one of the following values: <ul style="list-style-type: none"><li>• "min"</li><li>• "max"</li><li>• "horizontal"</li><li>• "vertical"</li></ul>
<code>canvas</code>	Canvas	R	The <code>Canvas</code> in which the event took place.
<code>currentTool</code>	string	R	The name of the current tool.
<code>far</code>	number	R	The distance from the <code>Camera</code> to the far clipping plane. A value of -1 for both <code>near</code> and <code>far</code> signifies to use auto clipping plane calculations.
<code>fov</code>	number	R	The size of the field of view for perspective <code>Camera</code> objects, measured in radians.
<code>isNewCanvas</code>	Boolean	R	Deprecated Determines whether this is the first event for this <code>Canvas</code> .
<code>near</code>	number	R	The distance from the <code>Camera</code> to the near clipping plane. A value of -1 for both <code>near</code> and <code>far</code> signifies to use auto clipping plane calculations.
<code>projectionType</code>	string	R	The type of projection, which can take one of the following values: <ul style="list-style-type: none"><li>• "perspective"</li><li>• "orthographic"</li></ul>
<code>targetDistance</code>	number	R	The distance from the <code>Camera</code> to its target.
<code>transform</code>	Matrix4x4	R	The <code>Camera</code> object's transformation matrix.
<code>viewPlaneSize</code>	number	R	The size of the view plane, measured in scene units.



# CameraEventHandler

Exposes a callback mechanism that allows a function to be evaluated when an camera event occurs. Event handlers are registered with the Runtime [addEventListener](#) method.

## CameraEventHandler

A constructor that creates a new `CameraEventHandler` object.

### Syntax

```
new CameraEventHandler ()
```

### Returns

A `CameraEventHandler` object.

## onEvent

A method that is called when a view is selected from the list of views on the 3D toolbar or in the context menu for an active 3D annotation.

### syntax

```
onEvent (event)
```

### Parameters

event	A <code>CameraEvent</code> object representing the event.
-------	---

### Returns

undefined

# Canvas

Represents a rectangular region into which the `Scene` is rendered from the viewpoint of the attached `Camera`.

See related objects, [Scene](#) and [Camera](#).

## Properties

Property	Type	Access	Description
<code>background</code>	Background	R	The Background object associated with the Canvas.

## getCamera

Obtains the `Camera` object attached to the `Canvas`.

## Syntax

```
getCamera ()
```

## Returns

A `Camera` object.

## setCamera

Sets the `Camera` object attached to the `Canvas`.

## Syntax

```
setCamera (camera)
```

## Parameters

<code>camera</code>	The <code>Camera</code> object used to set the object's value.
---------------------	--

## Returns

`undefined`

# ClippingPlane

An object representing a plane, within the `Scene`, that clips all geometry on one side of it. It is created by invoking the [createClippingPlane](#) method of the `Scene` object.

## remove

Removes the `ClippingPlane` object from the `Scene`.

## Syntax

```
remove ()
```

## Returns

undefined

# Color

An object that represents a RGB encoded color.

## Properties

Property	Type	Description
b	number	The blue component, which can be a value from 0 . 0 to 1 . 0 .
g	number	The green component, which can be a value from 0 . 0 to 1 . 0 .
r	number	The red component, which can be a value from 0 . 0 to 1 . 0 .

## Color

A constructor that creates a `Color` object, initialized to black.

## Syntax

```
new Color()
```

## Returns

A `Color` object, initialized to black.

## Color

A constructor that creates a `Color` object, initialized to the supplied RGB values.

## Syntax

```
new Color(r, g, b)
```

## Parameters

r	The red component, which can be a value from 0 . 0 to 1 . 0 .
g	The green component, which can be a value from 0 . 0 to 1 . 0 .
b	The blue component, which can be a value from 0 . 0 to 1 . 0 .

## Returns

A `Color` object, initialized to the supplied RGB values.

## set

Sets the `Color` object's value using an existing `Color` object

## Syntax

```
set (color)
```

## Parameters

color	The <code>Color</code> object used to set the object's value.
-------	---

## Returns

undefined

## set

Acrobat 7.0.7

Sets the `Color` object's value using the given RGB components.

## Syntax

```
set (r, g, b)
```

## Parameters

r	The red component, which can be a value from 0.0 to 1.0.
g	The green component, which can be a value from 0.0 to 1.0.
b	The blue component, which can be a value from 0.0 to 1.0.

## Returns

undefined

## set3

Deprecated

Sets the `Color` object's value using the given RGB components.

## Syntax

```
set3 (r, g, b)
```

## Parameters

r	The red component, which can be a value from 0.0 to 1.0.
g	The green component, which can be a value from 0.0 to 1.0.
b	The blue component, which can be a value from 0.0 to 1.0.

## Returns

undefined

# Console

This object can direct output to the JavaScript console in Acrobat for debugging purposes. The variable `console` is a global reference to this object.

## print

Prints a string to the JavaScript Console.

### Syntax

```
print(string)
```

### Parameters

<code>string</code>	The text to be printed to the JavaScript Console.
---------------------	---

### Returns

undefined

## println

Prints a string with an accompanying newline to the JavaScript Console.

### Syntax

```
println(string)
```

### Parameters

<code>string</code>	The text to be printed to the JavaScript Console.
---------------------	---

### Returns

undefined

## Dummy

Deprecated

A `Node` object used as an empty placeholder or a group within a `Scene`.



# FlashEvent

Acrobat 9.0

An object that is passed as an argument to the [onEvent](#) method of the FlashEventHandler object.

## Properties

Property	Type	Access	Description
command	string	R	<p>For a <code>FlashEvent</code> of type "command", this is the string representation of the command that has been sent through the <code>ActionScript FSCommand</code> function or through the <code>ExternalInterface.call</code> method.</p> <p>To execute the command, run the JavaScript function <code>eval</code> with the command string as an argument.</p>
target	FlashMovie	R	The target <code>FlashMovie</code> that the <code>FlashEvent</code> is from.
type	string	R	The type of <code>FlashEvent</code> , which can be "command", "progress", or "stateChange".
TYPE_COMMAND	string	R	A string constant for the <code>FlashEvent</code> type of "command".
TYPE_PROGRESS	string	R	A string constant for the <code>FlashEvent</code> type of "progress".
TYPE_STATECHANGE	string	R	A string constant for the <code>FlashEvent</code> type of "stateChange".
value	integer	R	<p>The value for the corresponding type of <code>FlashEvent</code>. The interpretation of <code>value</code> depends on the event type, "progress" or "stateChange".</p> <p><b>"progress"</b>: <code>value</code> is an integer from 0 to 100 representing the load progress of the <code>FlashMovie</code>.</p> <p><b>"stateChange"</b>: <code>value</code> is an integer signifying the ready state of the <code>FlashMovie</code>. Permitted values are 0 (Loading), 1 (Uninitialized), 2 (Loaded), 3 (Interactive), 4 (Complete).</p>

# FlashEventHandler

Acrobat 9.0

An object that exposes a callback mechanism that allows a function to be evaluated when an event occurs in a `FlashMovie` object. Event handlers are registered with the Runtime.[addEventListener](#) method.

## Properties

Property	Type	Access	Description
target	FlashMovie	R/W	When set, the <code>FlashEventHandler</code> will only report <code>FlashEvents</code> from the provided target <code>FlashMovie</code> .

## onEvent

A method that is called when an `ExternalInterface.call` method or `MMExecute` command is invoked from the `FlashMovie`'s `ActionScript`.

## Syntax

```
onEvent (event)
```

## Parameters

event	A <code>FlashEvent</code> object representing the event.
-------	--

## Returns

undefined

## FlashEventHandler

The constructor that creates a new `FlashEventHandler`.

## Syntax

```
new FlashEventHandler()
```

## Returns

A `FlashEventHandler` object.

# FlashMovie

Acrobat 9.0

An object that represents a Flash movie in the `Scene`.

## Properties

Property	Type	Access	Description
<code>alignMode</code>	integer	R/W	A bit flag that sets the alignment of the movie within the <code>Scene</code> . Values are +1 (left aligned), +2 (right aligned), +4 (top aligned), and +8 (bottom aligned).
<code>ALIGN_MODE_BOTTOM</code>	string	R	A string constant for the <code>FlashMovie</code> <code>scaleMode</code> type of "bottom".
<code>ALIGN_MODE_LEFT</code>	string	R	A string constant for the <code>FlashMovie</code> <code>scaleMode</code> type of "left".
<code>ALIGN_MODE_RIGHT</code>	string	R	A string constant for the <code>FlashMovie</code> <code>scaleMode</code> type of "right".
<code>ALIGN_MODE_TOP</code>	string	R	A string constant for the <code>FlashMovie</code> <code>scaleMode</code> type of "top".
<code>backgroundColor</code>	integer	R/W	<p>Override the background color of a movie. An integer of the form (<i>red</i> * 65536 + <i>green</i> * 256 + <i>blue</i>). Use a value of -1 for the default movie color.</p> <p>The values for <i>red</i>, <i>green</i> and <i>blue</i> are integers between 0 and 255, inclusive, and represent the color components of red, green, and blue, respectively, in the RGB color model.</p>
<code>desiredResolutionX</code>	integer	R/W	The desired resolution width for the <code>FlashMovie</code> content to be rendered at.
<code>desiredResolutionY</code>	integer	R/W	The desired resolution height for the <code>FlashMovie</code> content to be rendered at.
<code>frameNum</code>	integer	R/W	The frame number of the currently displayed frame of the movie. Setting this property advances or rewinds the movie.
<code>hitEnabled</code>	Boolean	R/W	Determines whether mouse events travel through the <code>FlashMovie</code> to elements in the scene behind it. If <code>true</code> , mouse events are trapped.

Property	Type	Access	Description
id	integer	R	A unique ID for each FlashMovie in the scene.
loop	Boolean	R/W	A flag that determines whether the animation loops. If <code>true</code> , the animation loops. If <code>false</code> , it plays only once.
opacity	number	R/W	The opacity of the FlashMovie represented by a value from 0.0 to 1.0, where 1.0 is completely opaque.
percentLoaded	integer	R	The percent of the Adobe Flash Player movie that has streamed into the browser so far with possible values from 0 to 100.
playing	Boolean	R	A flag that detects whether the movie is currently playing. If <code>true</code> , it is playing. If <code>false</code> , it is paused.
quality	integer	R/W	The current rendering quality. Permitted values are 0 (Low), 1 (High), 2 (AutoLow), and 3 (AutoHigh).
readyState	integer	R	The state of the FlashMovie. Permitted values are 0 (Loading), 1 (Uninitialized), 2 (Loaded), 3 (Interactive), 4 (Complete).
resolutionType	string	R/W	A string value that specifies the type of resolution to be used for the movie. Recognized values are "custom", "movie", and "window".
RESOLUTION_TYPE_CUSTOM	string	R	A string constant for the FlashMovie resolution type of "custom".
RESOLUTION_TYPE_MOVIE	string	R	A string constant for the FlashMovie resolution type of "movie".
RESOLUTION_TYPE_WINDOW	string	R	A string constant for the FlashMovie resolution type of "window".
scaleMode	string	R/W	The scale mode of the movie. The value of this property may be "exact fit", "no border", or "show all".
SCALE_MODE_EXACT_FIT	string	R	A string constant for the FlashMovie scaleMode type of "exact fit".
SCALE_MODE_NO_BORDER	string	R	A string constant for the FlashMovie's scaleMode type of "no border".

Property	Type	Access	Description
SCALE_MODE_SHOW_ALL	string	R	A string constant for the FlashMovie <code>scaleMode</code> type of "show all".
<code>totalFrames</code>	integer	R	The total number of frames in the movie. This is not available until the movie has loaded. Wait for <code>ReadyState = 4</code> .
<code>x</code>	integer	R/W	The x-position of the FlashMovie in the Canvas. Applies only to a FlashMovie if it is attached to the Foreground.
<code>y</code>	integer	R/W	The y-position of the FlashMovie in the Canvas. Applies only to a FlashMovie if it is attached to the Foreground.

## FlashMovie

Creates a new FlashMovie from a Resource of type "flash".

### Syntax

```
FlashMovie (FlashMovieResource)
```

### Parameters

FlashMovieResource	A Resource of type "flash".
--------------------	-----------------------------

### Returns

A FlashMovie object.

## call

Calls into `ActionScript` with the `ExternalInterface` calling convention to an exposed method (`ExternalInterface.addCallback` in `ActionScript`). The `call` method returns the return value of the method specified as the first parameter.

**Note:** The [JavaScript for Acrobat API Reference](#) has the `callAS` method of the [AnnotRichMedia](#) object that uses the same mechanism as the `call` method.

### Syntax

```
call (functionName, [argument1[, ...,argumentn]])
```

## Parameters

<code>functionName</code>	A string representing the function name to call in the <code>FlashMovie</code> <code>ActionScript</code> engine.
<code>argument1, argument2, ..., argumentn</code>	A comma-delimited list of arguments of varying type to be passed to the function in <code>ActionScript</code> .

## Returns

The return value from the called function, which can be of any type.

## getVariable

A method that returns the value of the Flash variable specified by `varName`, and returns `undefined` if the variable does not exist.

## Syntax

```
getVariable (varName)
```

## Parameters

<code>varName</code>	A string representing the variable requested.
----------------------	---

## Returns

A string representing the value of the specified Flash variable, or `undefined`.

## gotoFrame

Activates the frame number specified by `frameNumber` in the current movie. If the data for a requested frame is not yet available, the player goes to the last frame available and stops, causing unexpected results during playback. Use the `percentLoaded` property to determine if enough of the movie is available to execute the `gotoFrame()` method. The argument `frameNumber` is zero-based; that is, `frameNumber` is 0 in the first frame of the movie, 1 for the second frame, and so on. This differs from the `Goto` action within Flash, which begins at 1.

## Syntax

```
gotoFrame (frameNumber)
```

## Parameters

<code>frameNumber</code>	An integer representing the frame number.
--------------------------	---

## Returns

`undefined`

## isPlaying

A method that returns `true` if the movie is currently playing.

### Syntax

```
isPlaying()
```

### Returns

A Boolean type, `true` if the movie is playing, `false` otherwise.

## pan

This method pans a zoomed-in movie to the coordinates specified by `x` and `y`. Use `mode` to specify whether the values for `x` and `y` are pixels or a percentage of the window. The `pan` method does not pan beyond the boundaries of the zoomed-in movie.

### Syntax

```
pan(x, y, mode)
```

### Parameters

<code>x</code>	An integer representing the x coordinate.
<code>y</code>	An integer representing the y coordinate.
<code>mode</code>	When <code>mode</code> is 0, the coordinates are pixels; when <code>mode</code> is 1, the coordinates are a percentage of the window.

### Returns

undefined

## play

Starts playing the movie.

### Syntax

```
play()
```

### Returns

undefined

## rewind

Goes to the first frame.

## Syntax

```
rewind()
```

## Returns

undefined

## setVariable

Sets the value of the Flash variable specified by `variableName` to the value specified by `value`.

## Syntax

```
setVariable(varName, value)
```

## Parameters

<code>varName</code>	A string representing the variable requested.
<code>value</code>	A string value to be set for the provided variable name.

## Returns

undefined

## setZoomRect

Zooms in on a rectangular area of the movie. The units of the coordinates are measured in twips (1440 units per inch).

**Note:** To calculate the dimensions of a rectangle in the correct units, set the ruler units to Points and multiply the coordinates by 20 to get twips. (There are 72 points per inch.)

## Syntax

```
setZoomRect(left, top, right, bottom)
```

## Parameters

<code>left</code>	An integer representing the left side of the rectangle.
<code>top</code>	An integer representing the top side of the rectangle.
<code>right</code>	An integer representing the right side of the rectangle.
<code>bottom</code>	An integer representing the bottom side of the rectangle.

## Returns

undefined



## stop

Stops playing the movie.

### Syntax

```
stop()
```

### Returns

undefined

## zoom

This method zooms the view by a relative scale factor specified by percentage. For example, `zoom(50)` doubles the size of the objects in the view, `zoom(200)` reduces the size of objects in the view by one half, and `zoom(0)` resets the view to 100%. You cannot specify a scale factor that will zoom-out the original content further than 100%.

### Syntax

```
zoom(percentage)
```

### Parameters

percentage	An integer representing the zoom factor.
------------	--

### Returns

undefined

## HitInfo

The object returned when a hit test occurs during a [MouseEvent](#).

### Properties

Property	Type	Access	Description
distance	number	R	The distance from the Camera to the HitInfo object's position.
material	material	R	Acrobat 8.1 The material of the node that was hit.
position	vector3	R	The position of the point where the hit occurred.
surfaceNormal	vector3	R	Acrobat 8.1 The normal direction at the hit location on the world-space surface.
target	node	R	The target of the hit test.
textureCoordinate	vector3	R	Acrobat 8.1 The texture coordinate of the material that was hit.

## Host

Acrobat 7.0.7

An object that provides access to the JavaScript engine context and to pertinent objects within it. The variable `host` is a global reference to this object. It is a reference to the JavaScript Doc object in which the 3D annotation is contained.

# Image

An object that represents an image.

## Properties

Property	Type	Access	Description
height	number	R	The image's height, measured in pixels.
width	number	R	The image's width, measured in pixels.

## Image

A constructor that creates an new Image object.

## Syntax

```
new Image(resource)
```

## Parameters

resource	An Image object used to create the new object.
----------	--

## Returns

An Image object.

See [Image](#) for more information about the return object.

# KeyEvent

An object that is passed as an argument to the [onEvent](#) method of the `KeyEventHandler` object.

## Properties

Property	Type	Access	Description
<code>canvas</code>	<code>canvas</code>	R	The <code>Canvas</code> in which the <code>KeyEvent</code> took place.
<code>canvasPixelHeight</code>	<code>integer</code>	R	The height, measured in pixels, of the <code>Canvas</code> .
<code>canvasPixelWidth</code>	<code>integer</code>	R	The width, measured in pixels, of the <code>Canvas</code> .
<code>characterCode</code>	<code>integer</code>	R	The value of the character pressed according to Acrobat's character mapping, as per this listing of Acrobat character codes:

#	Keys	#	Keys	#	Keys
		65	A	97	a
		66	B	98	b
		67	C	99	c
		68	D	100	d
28	Left	69	E	101	e
29	Right	70	F	102	f
30	Down	71	G	103	g
31	Up	72	H	104	h
		73	I	105	i
		74	J	106	j
32	Space	75	K	107	k
		76	L	108	l
		77	M	109	m
48	0	78	N	110	n
49	1	79	O	111	o
50	2	80	P	112	p
51	3	81	Q	113	q
52	4	82	R	114	r
53	5	83	S	115	s
54	6	84	T	116	t
55	7	85	U	117	u
56	8	86	V	118	v
57	9	87	W	119	w
		88	X	120	x
		89	Y	121	y
		90	Z	122	z

Property	Type	Access	Description
ctrlKeyDown	Boolean	R	Determines whether the Ctrl key (Windows) or Command key (Mac OS) was pressed.  <b>Note:</b> Acrobat intercepts many of the Ctrl + key events because they are used for accelerators in the main application.
currentTool	string	R	The name of the current tool.
shiftKeyDown	Boolean	R	Determines whether the Shift key was pressed.  <b>Note:</b> Holding down the shift key changes the value of the <code>KeyEvent.characterCode</code> property.

# KeyEventHandler

An object that exposes a callback mechanism that allows a function to be evaluated when a key event occurs. Event handlers are registered with the Runtime [addEventListener](#) method.

## KeyEventHandler

A constructor that creates a new `KeyEventHandler` object.

### Syntax

```
new KeyEventHandler ()
```

### Returns

A `KeyEventHandler` object.

## onEvent

A method that is called when a key is pressed.

### Syntax

```
onEvent (event)
```

### Parameters

---

event	A <code>KeyEvent</code> object representing the event.
-------	--

---

### Returns

undefined

# Light

A `Node` object that illuminates meshes in the `Scene`.

There are different types of `Light` objects, each with their own distinct behavior. Infinite `Light` objects behave much like sunlight in that they cast parallel light in a given direction. Spot `Light` objects have a fixed cone angle that limits their beam to a conical projection. Point `Light` objects act similarly to a light bulb, where the light comes from a specific location in 3D space. Currently, none of the `Light` objects cast shadows.

In addition to the methods and properties that follow, the `Light` object also contains the same methods and properties as a `Node`.

## Properties

Property	Type	Access	Description
<code>attenuationA</code>	number	R/W	The <i>a</i> coefficient for <code>attenuationType</code> "abc".
<code>attenuationB</code>	number	R/W	The <i>b</i> coefficient for <code>attenuationType</code> "abc".
<code>attenuationC</code>	number	R/W	The <i>c</i> coefficient for <code>attenuationType</code> "abc".
<code>attenuationType</code>	string	R/W	The style of attenuation for the <code>Light</code> object being represented. Attenuation determines how fast the light intensity decreases with distance. The attenuation type of "abc" uses the equation $1 / \max( (a + bd + cdd), 1 )$ to determine the intensity where <i>d</i> is the distance from the light. One of the following values may be assigned: <ul style="list-style-type: none"> <li>"abc".</li> <li>"none"</li> </ul>
<code>ATTENUATION_ABC</code>	string	R	Acrobat 7.0.7 A string constant for the <code>attenuationType</code> of "abc".
<code>ATTENUATION_NONE</code>	string	R	Acrobat 7.0.7 A string constant for the <code>attenuationType</code> of "none".
<code>brightness</code>	number	R/W	Specifies the brightness of the emission from the <code>Light</code> . A value of 1 represents a brightness of 100%, though the property may be assigned higher values.
<code>color</code>	Color	R	Specifies the color of the light.

Property	Type	Access	Description
<code>direction</code>	Vector3	R	The direction toward which the light is pointing.
<code>directionLocal</code>	Vector3	R	Acrobat 7, but not documented until Acrobat 8.1  The direction toward which the light is pointing relative to its parent <code>Node</code> .
<code>innerConeAngle</code>	number	R/W	The angle, measured in radians, about the <code>direction</code> in which the light is of uniform full density.
<code>innerRadius</code>	number	R/W	The distance within which the light is of uniform full density.
<code>outerConeAngle</code>	number	R/W	The angle, measured in radians, about the <code>direction</code> outside of which the light's intensity is zero.
<code>outerRadius</code>	number	R/W	The distance beyond which the light's intensity is zero.
<code>position</code>	Vector3	R	The position of the <code>Light</code> object.
<code>positionLocal</code>	Vector3	R	The position of the <code>Light</code> object relative to its parent <code>Node</code> .
<code>type</code>	string	R/W	The type of <code>Light</code> object being represented. One of the following values may be assigned: <ul style="list-style-type: none"> <li>• "point"</li> <li>• "spot"</li> <li>• "infinite"</li> </ul>
<code>TYPE_INFINITE</code>	string	R	Acrobat 7.0.7  A string constant for the <code>Light</code> type of "infinite".
<code>TYPE_POINT</code>	string	R	Acrobat 7.0.7  A string constant for the <code>Light</code> type of "point".
<code>TYPE_SPOT</code>	string	R	Acrobat 7.0.7  A string constant for the <code>Light</code> type of "spot".



# Material

A `SceneObject` that controls the appearance of materials using the fixed function shader. In addition to the properties below, it contains the same methods and properties as a [SceneObject](#).

## Properties

Property	Type	Access	Description
<code>ambientColor</code>	Color	R	The ambient color.
<code>ambientTexture</code>	Texture	R	The ambient texture.
<code>bumpTexture</code>	Texture	R	A texture map whose value is used to describe the roughness of the object.
<code>diffuseColor</code>	Color	R	The matte color of an object.
<code>diffuseTexture</code>	Texture	R	A texture map that is used for the matte color of the object.
<code>emissiveColor</code>	Color	R	Emissive color except that it is does not require that any lighting to display. An object with an emissive color of white and no texture will appear pure white in the scene.
<code>emissiveTexture</code>	Texture	R	The emissive texture. Emissive texture is similar to ambient color, except that it is does not require that any lighting to display. An object with an emissive color of white and no texture will appear pure white in the scene.
<code>opacity</code>	number	R/W	The total opacity of the material.
<code>opacityTexture</code>	Texture	R	A texture map whose brightness is used for the level of opacity of the object. White signifies completely opaque while black signifies completely transparent.
<code>phongExponent</code>	number	R/W	The Phong exponent. The Phong exponent defines the “tightness” of the highlight. A higher exponent results in a smaller, tighter highlight while a lower exponent results in a broader flatter one.
<code>reflectionStrength</code>	number	R/W	The reflection level, which can be a value from 0.0 to 1.0.
<code>reflectionTexture</code>	Texture	R	The reflection texture, also known as an environment map, the texture is used to store the image of the environment surrounding the rendered object. It simulates the reflection of a mirrored surface

Property	Type	Access	Description
specularColor	Color	R	The specular color. The specularColor is the color of the highlight on the material.
specularStrength	number	R/W	The specular strength, which is a measure of how shiny the material is.

## attachFlashMovie

Acrobat 9.0

Sets the provided `FlashMovie` as the diffuse texture for the `Material`.

### Syntax

```
attachFlashMovie(movie)
```

### Parameters

<code>movie</code>	The <code>FlashMovie</code> object to be used as the diffuse texture.
--------------------	---

### Returns

undefined

## Matrix4x4

A four-by-four matrix commonly used for transformations.

### Properties

Property	Type	Access	Description
determinant	number	R/W	The determinant of the matrix.
inverse	Matrix4x4	R	The inverse of the matrix.
scaleComponent	Vector3	R	The scale component of the transformation.
translation	Vector3	R	The translation component of the transformation.
transpose	Matrix4x4	R	The transpose of the matrix.

## Matrix4x4

A constructor that creates a new `Matrix4x4` object initialized to the identity matrix.

### Syntax

```
new Matrix4x4()
```

### Returns

A `Matrix4x4` object initialized to the identity matrix.

## Matrix4x4

A constructor that creates a new `Matrix4x4` object initialized to the specified matrix.

### Syntax

```
new Matrix4x4(matrix)
```

### Parameters

<code>matrix</code>	A <code>Matrix4x4</code> object used to initialize the new matrix.
---------------------	--

### Returns

A `Matrix4x4` object initialized to the specified matrix.

## invertInPlace

Inverts the matrix.

### Returns

undefined

## isEqual

Determines whether the current matrix is equal to the specified matrix.

### Syntax

```
isEqual(matrix)
```

### Parameters

---

<code>matrix</code>	A <code>Matrix4x4</code> object used for the comparison.
---------------------	--

---

### Returns

Returns `true` if the matrices are equal, `false` otherwise.

## multiply

Multiplies the current matrix by the specified matrix.

### Syntax

```
multiply(matrix)
```

### Parameters

---

<code>matrix</code>	A <code>Matrix4x4</code> object used for the multiplication.
---------------------	--

---

### Returns

A `Matrix4x4` object.

## multiplyInPlace

Multiplies the current matrix by the specified matrix, and updates the current matrix with the resulting value.

### Syntax

```
multiplyInPlace(matrix)
```

## Parameters

---

<code>matrix</code>	A <code>Matrix4x4</code> object used for the multiplication.
---------------------	--

---

## Returns

undefined

## rotateWithQuaternion

Rotates the current matrix using the specified `Quaternion`.

## Syntax

```
rotateWithQuaternion( quaternion )
```

## Parameters

---

<code>quaternion</code>	A <code>Quaternion</code> object used for the rotation.
-------------------------	---

---

## Returns

A `Matrix4x4` object.

## rotateWithQuaternionInPlace

Rotates the current matrix using the specified quaternion, and updates the current matrix with the resulting value.

## Syntax

```
rotateWithQuaternionInPlace( quaternion )
```

## Parameters

---

<code>quaternion</code>	A <code>Quaternion</code> object used for the rotation.
-------------------------	---

---

## Returns

undefined

## rotateAboutLine

Rotates the current matrix about the specified line.

## Syntax

```
rotateAboutLine( angle, start, end )
```

## Parameters

angle	The angle of rotation, in radians.
start	A point described by a <code>Vector3</code> object used to specify the beginning of the line of rotation, which is represented by <code>start - end</code> .
end	A point described by a <code>Vector3</code> object used to specify the end of the line of rotation, which is represented by <code>start - end</code> .

## Returns

A `Matrix4x4` object.

## rotateAboutLineInPlace

Rotates the current matrix about the specified line, and updates the current matrix with the resulting value.

## Syntax

```
rotateAboutLineInPlace (angle, start, end)
```

## Parameters

angle	The angle of rotation, in radians.
start	A <code>Vector3</code> object used to specify the line of rotation, which is represented by <code>start - end</code> .
end	A <code>Vector3</code> object used to specify the line of rotation, which is represented by <code>start - end</code> .

## Returns

undefined

## rotateAboutX

Rotates the current matrix about the x axis.

## Syntax

```
rotateAboutX (angle)
```

## Parameters

angle	The angle of rotation, in radians.
-------	------------------------------------

## Returns

A `Matrix4x4` object.

## rotateAboutXInPlace

Rotates the current matrix about the x axis, and updates the current matrix with the resulting value.

## Syntax

```
rotateAboutXInPlace (angle)
```

## Parameters

angle	The angle of rotation, in radians.
-------	------------------------------------

## Returns

undefined

## rotateAboutVector

Rotates the current matrix about the specified vector.

## Syntax

```
rotateAboutVector (angle, axis)
```

## Parameters

angle	The angle of rotation, in radians.
axis	A <code>Vector3</code> object about which the matrix is rotated.

## Returns

A `Matrix4x4` object.

## rotateAboutVectorInPlace

Rotates the current matrix about the specified vector, and updates the current matrix with the resulting value.

## Syntax

```
rotateAboutVectorInPlace (angle, axis)
```

## Parameters

angle	The angle of rotation, in radians.
axis	A <code>Vector3</code> object about which the matrix is rotated.

## Returns

undefined

## rotateAboutY

Rotates the current matrix about the y axis.

## Syntax

```
rotateAboutY (angle)
```

## Parameters

angle	The angle of rotation, in radians.
-------	------------------------------------

## Returns

A `Matrix4x4` object.

## rotateAboutYInPlace

Rotates the current matrix about the y axis, and updates the current matrix with the resulting value.

## Syntax

```
rotateAboutYInPlace (angle)
```

## Parameters

angle	The angle of rotation, in radians.
-------	------------------------------------

## Returns

undefined

## rotateAboutZ

Rotates the current matrix about the z axis.



## Syntax

```
rotateAboutZ (angle)
```

## Parameters

angle	The angle of rotation, in radians.
-------	------------------------------------

## Returns

A `Matrix4x4` object.

## rotateAboutZInPlace

Rotates the current matrix about the z axis, and updates the current matrix with the resulting value.

## Syntax

```
rotateAboutZInPlace (angle)
```

## Parameters

angle	The angle of rotation, in radians.
-------	------------------------------------

## Returns

undefined

## scale

Scales the current matrix using the specified scaling components.

## Syntax

```
scale (x, y, z)
```

## Parameters

x	The scaling component in the x direction.
y	The scaling component in the y direction.
z	The scaling component in the z direction.

## Returns

A `Matrix4x4` object.

## scaleInPlace

Scales the current matrix using the specified scaling components, and updates the current matrix with the resulting value.

### Syntax

```
scaleInPlace(x, y, z)
```

### Parameters

x	The scaling component in the x direction.
y	The scaling component in the y direction.
z	The scaling component in the z direction.

### Returns

undefined

## set

Sets the value of the current matrix using the specified matrix.

### Syntax

```
set(matrix)
```

### Parameters

matrix	The matrix whose value is copied into the current matrix.
--------	---

### Returns

undefined

## set

Acrobat 8.1

Sets the value of the current matrix using an array.

### Syntax

```
set( array )
```

## Parameters

---

array	The array of length 16 whose values are copied into the current matrix.
-------	---

---

## Returns

undefined

## set

Acrobat 8.1

Sets the value of the current matrix using 16 numeric values.

## Syntax

```
set(v0, v1, v2, v3, v4, v5, v6, v7, v8, v9, v10, v11, v12, v13, v14, v15)
```

## Parameters

---

v0-v15	Number values for the given indices of the matrix.
--------	--

---

## Returns

undefined

## setIdentity

Sets the value of the current matrix to the identity matrix.

## Syntax

```
setIdentity()
```

## Returns

undefined

## setView

Sets the current matrix according to the specified component vectors.

## Syntax

```
setView(position, direction, up)
```

## Parameters

position	A <code>Vector3</code> object used to specify the position component.
direction	A <code>Vector3</code> object used to specify the direction component.
up	A <code>Vector3</code> object used to specify the upward component.

## Returns

undefined

## transformDirection

Transforms the specified vector by the current matrix.

## Syntax

```
transformDirection(vector)
```

## Parameters

vector	The <code>Vector3</code> object to be transformed.
--------	--

## Returns

A `Vector3` object.

## transformPosition

Transforms the specified position by the current matrix.

## Syntax

```
transformPosition(position)
```

## Parameters

position	A <code>Vector3</code> object representing the position to be transformed.
----------	--

## Returns

A `Vector3` object.

## translate

Translates the current matrix by the components of the specified vector.

## Syntax

```
translate(translation)
```

## Parameters

---

translation	The <code>Vector3</code> object whose components are used to perform the matrix translation.
-------------	--

---

## Returns

A `Matrix4x4` object.

## translateInPlace

Translates the current matrix by the components of the specified vector, and updates the current matrix with the resulting value.

## Syntax

```
translateInPlace(translation)
```

## Parameters

---

translation	The <code>Vector3</code> object whose components are used to perform the matrix translation.
-------------	--

---

## Returns

undefined

## transposeInPlace

Sets the value of the current matrix to its transpose.

## Syntax

```
transposeInPlace()
```

## Returns

undefined

# MenuEvent

An object that is passed as an argument to the `onEvent` method of the `MenuEventHandler` object.

## Properties

Property	Type	Access	Description
<code>canvas</code>	Canvas	R	The Canvas in which the <code>MenuEvent</code> took place.
<code>currentTool</code>	string	R	The name of the current tool.
<code>menuItemChecked</code>	Boolean	R	Determines whether the menu item was selected.
<code>menuItemName</code>	string	R	The name of the selected menu item.

# MenuEventHandler

A `MenuEventHandler` object exposes a callback mechanism that allows a function to be evaluated when an event occurs. Event handlers are registered with the `Runtime.addEventHandler` method.

## MenuEventHandler

A constructor that creates a new `MenuEventHandler` object.

### Syntax

```
new MenuEventHandler ()
```

### Returns

A `MenuEventHandler` object.

## onEvent

A method that is called when a custom menu item is selected on the context menu for an active 3D annotation.

### Syntax

```
onEvent (event)
```

### Parameters

---

<code>event</code>	A <code>MenuEvent</code> object representing the event.
--------------------	---

---

### Returns

undefined

# Mesh

A `Node` object that contains geometry. A `Mesh` object with no geometry has children `Node` objects that can be transformed as a group. In addition to the methods and properties below, it contains the same methods and properties as a [Node](#).

## Properties

Property	Type	Description
<code>material</code>	<code>material</code>	The <code>Mesh</code> object's default <code>Material</code> .
<code>renderMode</code>	<code>string</code>	The <code>Mesh</code> object's rendering style, which can be one of the following values: <ul style="list-style-type: none"><li>• <code>"default"</code></li><li>• <code>"bounding box"</code></li><li>• <code>"transparent bounding box"</code></li><li>• <code>"transparent bounding box outline"</code></li><li>• <code>"vertices"</code></li><li>• <code>"shaded vertices"</code></li><li>• <code>"wireframe"</code></li><li>• <code>"shaded wireframe"</code></li><li>• <code>"solid"</code></li><li>• <code>"transparent"</code></li><li>• <code>"solid wireframe"</code></li><li>• <code>"transparent wireframe"</code></li><li>• <code>"illustration"</code></li><li>• <code>"solid outline"</code></li><li>• <code>"shaded illustration"</code></li><li>• <code>"hidden wireframe"</code></li></ul>

## computeBoundingBox

Acrobat 7.0.7

Computes the bounds of the `Node` object.

## Syntax

```
computeBoundingBox()
```

## Returns

A `BoundingBox` object.



# MouseEvent

An object that is passed as an argument to the `onEvent` method of the [MouseEventHandler](#) object.

## Properties

Property	Type	Access	Description
<code>canvas</code>	Canvas	R	The Canvas in which the <code>MouseEvent</code> took place.
<code>canvasPixelHeight</code>	integer	R	The height, measured in pixels, of the Canvas in which the <code>MouseEvent</code> took place.
<code>canvasPixelWidth</code>	integer	R	The width, measured in pixels, of the Canvas in which the <code>MouseEvent</code> took place.
<code>ctrlKeyDown</code>	Boolean	R	Determines whether the Ctrl key (Windows) or Command key (Mac OS) was pressed.
<code>currentTool</code>	string	R	The name of the current tool.
<code>hits</code>	Array	R	A set of <code>HitInfo</code> objects ordered by distance from nearest to furthest.
<code>isDoubleClick</code>	Boolean	R	Determines whether a double-click event occurred
<code>isMouseDown</code>	Boolean	R	Determines whether the mouse button was pressed
<code>isMouseHit</code>	Boolean	R	Determines whether the target is under the mouse cursor.
<code>isMouseMove</code>	Boolean	R	Determines whether the mouse position changed.
<code>isMouseOut</code>	Boolean	R	Determines whether the mouse position moved off the target.
<code>isMouseOver</code>	Boolean	R	Determines whether the mouse position moved onto the target.
<code>isMouseUp</code>	Boolean	R	Determines whether the mouse button was released.
<code>leftButtonDown</code>	Boolean	R	Determines whether the left mouse button was pressed.
<code>mouseX</code>	integer	R	The x position of the mouse cursor in the Canvas .
<code>mouseY</code>	integer	R	The y position of the mouse cursor in the Canvas .

Property	Type	Access	Description
rightButtonDown	Boolean	R	Version 7.0.1 Determines whether the right mouse button was pressed.
shiftKeyDown	Boolean	R	Determines whether the Shift key was pressed.

## MouseEventHandler

An object that exposes a callback mechanism that allows a function to be evaluated when a mouse event occurs. The handler may be customized to filter out certain event types. Event handlers are registered with the Runtime [addEventListener](#) method.

### Properties

Property	Type	Access	Description
onMouseDoubleClick	Boolean	R/W	When set to true, the handler is called back when a mouse button is clicked twice in rapid succession on the target object. If no target is specified, the handler calls back on any double-click.
onMouseDown	Boolean	R/W	When set to true, the handler is called back when a mouse button is initially pressed while the cursor is over the target object. If no target is specified, the handler calls back on any button press.
onMouseHit	Boolean	R/W	When set to true, the handler is called back continuously when the cursor is over the target object. In the case of onMouseHit, it does not matter if the target object is behind another object in the scene. The list of resultant hit objects are provided in the <code>MouseEvent.hits</code> property.
onMouseMove	Boolean	R/W	When set to true, the handler is called back when the cursor moves over the target object. If no target is specified, the handler calls back on any mouse movement over the 3D annotation.
onMouseOut	Boolean	R/W	When set to true, the handler is called back once when the cursor moves off of the target object. To be called back, the target must be the frontmost object. To exclude objects, use the <code>Node.hitEnabled</code> property.
onMouseOver	Boolean	R/W	When set to true, the handler is called back once when the cursor moves over the target object.
onMouseUp	Boolean	R/W	When set to true, the handler is called back when a mouse button is initially released. If a target is specified, it calls back only when the cursor is over the handler's target.

Property	Type	Access	Description
reportAllTargets	Boolean	R/W	Determines whether a hit test is performed. When set to <code>false</code> , a hit test is not performed except on a mouse-down or mouse-up event. This is an optimization feature because the current hit test is extremely expensive on complex models. When set to <code>false</code> , the following events are not reported because they depend on hit testing: <ul style="list-style-type: none"><li>• <code>mouse-hit</code></li><li>• <code>mouse-move</code></li><li>• <code>mouse-out</code></li><li>• <code>mouse-over</code></li></ul>
target	object	R/W	The Mesh or Background object on which the mouse event occurs.

## MouseEventHandler

A constructor that creates a new `MouseEventHandler` object.

### Syntax

```
new MouseEventHandler()
```

### Returns

A `MouseEventHandler` object.

## onEvent

A method that is called when a mouse event occurs.

### Syntax

```
onEvent (event)
```

### Parameters

event	A <code>MouseEvent</code> object representing the event.
-------	--

### Returns

undefined

## Node

An object within the Scene hierarchy (a `SceneObject`) that has a 3D representation. The following objects are considered `Node` objects:

- [Bone](#)
- [Camera](#)
- [ClippingPlane](#)
- [Dummy](#)
- [Light](#)
- [Mesh](#)
- [Procedural](#)

To obtain a `Node` object's type, use the standard JavaScript `constructor` property. For example, the following syntax prints the `Node` object's type to the console:

```
console.println(myNode.constructor.name);
```

In addition to the methods and properties below, it contains the same methods and properties as a [SceneObject](#).

### Properties

Property	Type	Access	Description
<code>firstChild</code>	<code>Node</code> (if the first child exists), <code>None</code> otherwise	R	The <code>Node</code> object's first child.
<code>hitEnabled</code>	<code>Boolean</code>	R/W	Determines whether the <code>Node</code> is included in hit tests. The default value is <code>true</code> .
<code>info</code>	<code>string</code>	R	Acrobat 7.0.7 Information associated with the <code>Node</code> .
<code>metadataString</code>	<code>string</code>	R	Acrobat 8.1 A string containing <code>Node</code> -specific metadata.
<code>nextSibling</code>	<code>Node</code> (if the next sibling exists), <code>None</code> otherwise	R	The next sibling.
<code>opacity</code>	<code>number</code>	R/W	Acrobat 7.0.7 The <code>Node</code> opacity. A value from 0 to 1, where 1 is completely opaque.
<code>parent</code>	<code>object</code>	R	The <code>Node</code> object's parent <code>Node</code> or <code>Scene</code> .
<code>transform</code>	<code>Matrix4x4</code>	R	The local to world transformation matrix for the <code>Node</code> .

Property	Type	Access	Description
wireframeColor	Color	R	The <code>Color</code> object used to determine the wireframe appearance.
visible	Boolean	R/W	Determines whether the <code>Node</code> object should be shown. This property applies to mesh nodes only. For example, modifying the empty parent node of a mesh tree has no effect on the child mesh tree items. In such cases it is recommended that you modify a parent node that is also a mesh node, and the child mesh items will have the same value for this property.

## detachFromCurrentAnimation

Removes the ability of the currently active `Animation` of the `Node` object to transform the `Node`.

### Syntax

```
detachFromCurrentAnimation()
```

### Returns

undefined

## Procedural

Deprecated

A `Node` object used to represent procedurally created geometry, such as constructive solid geometry (CSG) solids, procedural spheres, or NURB objects (a 3D curve or surface). A `Procedural` object contains the same methods and properties as a [Node](#).

# Quaternion

Represents a rotation in 3D space, and allows for smooth interpolation (blending) between orientations of subjects. A `Quaternion` is typically used for animating a `Camera` or `Mesh` over time, and can be converted to and from angles of rotation about the axes.

## Quaternion

A constructor that initializes the object with the identity matrix.

### Syntax

```
new Quaternion()
```

### Returns

A `Quaternion` object.

## Quaternion

A constructor that initializes the object with the specified rotation matrix.

### Syntax

```
new Quaternion(matrix)
```

### Parameters

---

<code>matrix</code>	A <code>Matrix4x4</code> object representing the rotation matrix.
---------------------	---

---

### Returns

A `Quaternion` object.

## Quaternion

A constructor that initializes the object with the specified `Quaternion`.

### Syntax

```
new Quaternion(Quaternion)
```

### Parameters

---

<code>Quaternion</code>	A <code>Quaternion</code> object used to initialize the new object.
-------------------------	---

---



## Returns

A Quaternion object.

## interpolate

Creates a Quaternion object interpolated from the current and specified Quaternion objects and a.

## Syntax

```
interpolate( quaternion, a )
```

## Parameters

quaternion	A Quaternion object used to interpolate the new object.
a	A number value, from 0.0 to 1.0, that specifies the degree (percentage) of interpolation. A value of 0.5 represents an interpolation halfway between the current and specified Quaternion objects.

## Returns

A Quaternion object.

## interpolateInPlace

Creates a Quaternion object interpolated from the current and specified Quaternion objects and a, and updates the current Quaternion object with the new value.

## Syntax

```
interpolateInPlace( quaternion, a )
```

## Parameters

quaternion	A Quaternion object used to interpolate the new object.
a	A number value, from 0.0 to 1.0, that specifies the degree (percentage) of interpolation. A value of 0.5 represents an interpolation halfway between the current and specified Quaternion objects.

## Returns

A Quaternion object.

## normalize

Normalizes the Quaternion object

## Syntax

`normalize()`

## Returns

`undefined`

## RenderEvent

An object that is passed as an argument to the `onEvent` method of the `RenderEventHandler` object .

### Properties

Property	Type	Access	Description
<code>canvas</code>	Canvas	R	The Canvas that is the target of the <code>RenderEvent</code> .
<code>canvasPixelHeight</code>	integer	R	The height, measured in pixels, of the Canvas for which the <code>RenderEvent</code> is intended.
<code>canvasPixelWidth</code>	integer	R	The width, measured in pixels, of the Canvas for which the <code>RenderEvent</code> is intended.
<code>currentTool</code>	string	R	The name of the current tool.

# RenderEventHandler

An object that exposes a callback mechanism that allows a function to be evaluated when an event occurs. Event handlers are registered with the Runtime [addEventHandler](#) method. It issues a callback just before each `Canvas` is rendered.

## RenderEventHandler

A constructor that creates a new `RenderEventHandler` object.

### Syntax

```
new RenderEventHandler ()
```

### Returns

A `RenderEventHandler` object.

## onEvent

A method that is called immediately before the `Canvas` is rendered.

### Syntax

```
onEvent (event)
```

### Parameters

event	A <code>RenderEvent</code> object representing the event.
-------	---

### Returns

undefined

## RenderOptions

An object that describes the style with which to render `Node` objects in the `Scene`.

### Properties

Property	Type	Access	Description
<code>boundingBoxColor</code>	Color	R	A <code>Color</code> object to be applied to the bounding box.
<code>clippingPlaneColor</code>	Color	R	A <code>Color</code> object to be applied to the clipping plane.
<code>clippingPlaneIntersectionColor</code>	Color	R	A <code>Color</code> object to be applied to the clipping plane intersection.
<code>defaultAmbientColor</code>	Color	R	A <code>Color</code> object to be applied to the default ambient <code>Material</code> .
<code>defaultDiffuseColor</code>	Color	R	A <code>Color</code> object to be applied to the default diffuse <code>Material</code> .
<code>defaultEmissiveColor</code>	Color	R	A <code>Color</code> object to be applied to the default emissive <code>Material</code> .
<code>defaultSpecularColor</code>	Color	R	A <code>Color</code> object to be applied to the default specular <code>Material</code> .
<code>illustrationRenderModeFaceColor</code>	Color	R	Acrobat 7.0.7 The color of the faces when the render mode is <code>Illustration</code> .
<code>illustrationRenderModeLineColor</code>	Color	R	A <code>Color</code> object to be applied to the illustration lines.
<code>pointsRenderModeColor</code>	Color	R	A <code>Color</code> object to be applied to the vertices in point render mode.
<code>shadedIllustrationRenderModeLineColor</code>	Color	R	A <code>Color</code> object to be applied to the shaded illustration lines.

Property	Type	Access	Description
<code>solidGridColorEven</code>	Color	R	Acrobat 7.0.7 The color of the even squares of the checkered grid when drawn in solid mode.
<code>solidGridColorOdd</code>	Color	R	Acrobat 7.0.7 The color of the odd squares of the checkered grid when drawn in solid mode.
<code>solidRenderModeLineColor</code>	Color	R	A <code>Color</code> object to be applied to the solid or transparent lines in render mode.
<code>transparentBoundsRenderModeColor</code>	Color	R	A <code>Color</code> object to be applied to the transparent bounding box.
<code>transparentGridColorEven</code>	Color	R	Acrobat 7.0.7 The color of the even squares of the checkered grid when drawn in transparent mode.
<code>transparentGridColorOdd</code>	Color	R	Acrobat 7.0.7 The color of the odd squares of the checkered grid when drawn in transparent mode.
<code>wireframeRenderModeColor</code>	Color	R	Acrobat 7.0.7 The color of the wires when the render mode is Wireframe.
<code>xAxisColor</code>	Color	R	Acrobat 7.0.7 The color of the x axis.
<code>yAxisColor</code>	Color	R	Acrobat 7.0.7 The color of the y axis.
<code>zAxisColor</code>	Color	R	Acrobat 7.0.7 The color of the z axis.

## Resource

An object that creates an abstraction for loading behavior in files and streams.

### Properties

Property	Type	Access	Description
type	string	R	The type of <code>Resource</code> object, which can be one of the following values: <ul style="list-style-type: none"><li>• "image"</li><li>• "model"</li><li>• "flash" (Acrobat 9.0)</li><li>• "unknown"</li></ul>
TYPE_IMAGE	string	R	Acrobat 7.0.7 A string constant for the <code>Resource</code> type of "image".
TYPE_MODEL	string	R	Acrobat 7.0.7 A string constant for the <code>Resource</code> type of "model".
TYPE_UNKNOWN	string	R	Acrobat 7.0.7 A string constant for the <code>Resource</code> type of "unknown".
TYPE_FLASH	string	R	Acrobat 9.0 A string constant for the <code>Resource</code> type of "flash".

## Resource

A constructor that creates a new `Resource`.

### Syntax

```
new Resource(pathname)
```

### Parameters

pathname	A string representing the path of the file or stream. Can load embedded resources only from within the PDF file. The pathname string must start with pdf://.
----------	--

### Returns

A `Resource` object.

## Runtime

An object that represents the run-time instance of the player. Each `Runtime` object can have its own unique script engine and set of annotations. The variable `runtime` is a global reference to this object.

### Properties

Property	Type	Access	Description
<code>BUTTON_TYPE_PUSH</code>	string	R	Acrobat 7.0.7 A string constant for the custom tool button type of push button. It is used with the <code>addCustomToolButton</code> method.
<code>BUTTON_TYPE_TOOL</code>	string	R	Acrobat 7.0.7 A string constant for the custom button type of tool button. It is used with the <code>addCustomToolButton</code> method.
<code>canvasCount</code>	number	R	Acrobat 8.1 The number of Canvases that are attached to the active 3D annotation.
<code>ctrlKeyDown</code>	Boolean	R	Determines whether the Ctrl key (Windows) or Command key (Mac OS) was pressed.
<code>eventHandlerCount</code>	integer	R	The number of registered event handlers.
<code>instances</code>	Array	R	Acrobat 7.0.7 An array of JavaScript <code>Annot3D</code> objects that are attached to the 3D script context.
<code>MENU_ITEM_TYPE_CHECKED</code>	string	R	Acrobat 7.0.7 A string constant for the custom menu item type of checked. It is used with the <code>addCustomMenuItem</code> method.
<code>MENU_ITEM_TYPE_DEFAULT</code>	string	R	Acrobat 7.0.7 A string constant for the custom menu item type of default. It is used with the <code>addCustomMenuItem</code> method.
<code>overrideFlyTool</code>	Boolean	R/W	Acrobat 9.0 Determines whether to override the built-in Fly tool behavior.
<code>overrideNavTools</code>	Boolean	R/W	Determines whether to disable all default navigation behavior. <b>Note:</b> Setting this property does not prevent view changes.



Property	Type	Access	Description
overridePanTool	Boolean	R/W	Determines whether to override the built-in Pan tool behavior.  <b>Note:</b> Setting this property does not affect the pan behavior of other navigation tools.
overrideRotateTool	Boolean	R/W	Determines whether to override the built-in Rotate tool behavior.
overrideSelection	Boolean	R/W	Acrobat 7.0.7 Determines whether to override the built-in Selection tool behavior.
overrideSpinTool	Boolean	R/W	Acrobat 8.0 Determines whether to override the built-in Spin tool behavior.
overrideViewChange	Boolean	R/W	Determines whether to override the setting of Views from Acrobat.
overrideWalkTool	Boolean	R/W	Determines whether to override the built-in Walk tool behavior.
overrideScrollWheel	Boolean	R/W	Acrobat 8.1 Determines whether to override the built-in scroll-wheel behavior.
overrideZoomTool	Boolean	R/W	Determines whether to override the built-in Zoom tool behavior.  <b>Note:</b> Setting this property does not affect the zoom behavior of other navigation tools.
scrollWheelSpeed	number	R/W	Acrobat 8.1 A speed multiplier for the value of the scroll-wheel motion.
shiftKeyDown	Boolean	R	Determines whether the Shift key was pressed.

Property	Type	Access	Description
speedThreshold	number	R/W	<p>Acrobat 8.1</p> <p>A length (based upon the diagonal of the scene's bounding box) under which the Walk tool's motion is scaled relative to the size of the model.</p> <p>The Walk tool's motion is constant based upon the scene's scale factor, such that it emulates a natural pace relative to the model's size. This works well for architectural models that are created with a defined scale. However, the walk motion is too quick for very small models.</p>
strafeSpeed	number	R/W	<p>Acrobat 8.1</p> <p>A speed multiplier for the lateral motion while using the Walk tool.</p>
TOOL_NAME_FLY	string	R	<p>Acrobat 9.0</p> <p>A string constant for the name of the fly tool. Its value is "Fly".</p>
TOOL_NAME_MEASURE	string	R	<p>Acrobat 7.0.7</p> <p>A string constant for the name of the measure tool. Its value is "Measure".</p>
TOOL_NAME_PAN	string	R	<p>Acrobat 7.0.7</p> <p>A string constant for the name of the pan tool. Its value is "Pan".</p>
TOOL_NAME_ROTATE	string	R	<p>Acrobat 7.0.7</p> <p>A string constant for the name of the rotate tool. Its value is "Rotate".</p>
TOOL_NAME_SPIN	string	R	<p>Acrobat 8.0</p> <p>A string constant for the name of the Spin tool. Its value is "Spin".</p>
TOOL_NAME_WALK	string	R	<p>Acrobat 7.0.7</p> <p>A string constant for the name of the walk tool. Its value is "Walk".</p>
TOOL_NAME_ZOOM	string	R	<p>Acrobat 7.0.7</p> <p>A string constant for the name of the zoom tool. Its value is "Zoom".</p>
version	number	R	<p>The number corresponding to the version of the Runtime system.</p>

Property	Type	Access	Description
viewCount	integer	R	Acrobat 9.0 The number of named views for the annotation.
walkSpeed	number	R/W	Acrobat 8.1 A speed multiplier for the forward/backward motion while using the Walk tool.

## addCustomMenuItem

Creates a custom menu item in the 3D annotation context menu.

### Syntax

```
addCustomMenuItem(name, label, type, checkedState)
```

### Parameters

name	A string identifying the menu item.
label	A string appearing on the menu item.
type	A string indicating whether it is a checked menu item. A checked menu item has a check mark toggle next to it. Its possible values are: <ul style="list-style-type: none"><li>"default"</li><li>"checked"</li></ul>
checkedState	A Boolean value indicating the state of a checked menu item.

### Returns

undefined

## addCustomToolButton

Creates a custom tool button in the 3D toolbar.

### Syntax

```
addCustomToolButton(name, label, type)
```

### Parameters

name	A string identifying the tool button.
------	---------------------------------------

---

label	A string appearing on the tool button.
type	A string indicating whether it is a tool button or a push button. Its possible values are: <ul style="list-style-type: none"><li>• "tool button"</li><li>• "push button"</li></ul>

---

## Returns

undefined

## addEventHandler

Registers the provided event handler.

## Syntax

```
addEventHandler(eventHandler)
```

## Parameters

---

eventHandler	The event handler object to be registered.
--------------	--

---

## Returns

undefined

## disableTool

Disables the tool with the specified ID.

## Syntax

```
disableTool(toolName)
```

## Parameters

---

toolName	A string identifying the tool.
----------	--------------------------------

---

## Returns

undefined

## enableTool

Enables the tool with the specified ID.

## Syntax

```
enableTool (toolName)
```

## Parameters

toolName	A string identifying the tool.
----------	--------------------------------

## Returns

undefined

## getEventHandler

Obtains the event handler corresponding to the specified index.

## Syntax

```
getEventHandler (index)
```

## Parameters

index	An integer identifying the event handler.
-------	---

## Returns

An event handler object.

## getRendererName

Obtains the name of the current renderer.

## Syntax

```
getRendererName ()
```

## Returns

A string containing the current renderer's name.

## getView

Acrobat 9.0

Gets the indicated view for the annotation by its index.

See the related method, [setView](#), for setting the view by its index.

## Syntax

```
getView( index )
```

## Parameters

index	The integer index of the view.
-------	--------------------------------

## Returns

View

## getView

Acrobat 9.0

Gets the indicated view for the annotation by its name.

See the related method, [setView](#), for setting the view by its name.

## Syntax

```
getView( name )
```

## Parameters

name	The string name of the view.
------	------------------------------

## Returns

View

## pause

Acrobat 9.0

Pauses the runtime. This is the same as selecting the Pause toolbar button or menu item.

## Syntax

```
pause ( )
```

## Returns

undefined

## play

Acrobat 9.0

Resumes playback of the runtime. This is the same as selecting the Play toolbar button or menu item.

## Syntax

```
play()
```

## Returns

undefined

## refresh

Version 7.0.1

Marks the render area dirty so that it is redrawn. This is useful when something changes in the scene but the annotation is in a Loaded and not Live state.

## Syntax

```
refresh()
```

## Returns

undefined

## removeEventHandler

Unregisters the specified event handler.

## Syntax

```
removeEventHandler(handler)
```

## Parameters

handler	An event handler object representing the event handler.
---------	---

## Returns

undefined

## removeCustomMenuItem

Removes the custom menu item with the specified ID.

## Syntax

```
removeCustomMenuItem(menuName)
```

## Parameters

---

menuName	A string identifying the custom menu item.
----------	--

---

## Returns

undefined

## removeCustomToolButton

Removes the custom tool button with the specified ID.

## Syntax

```
removeCustomToolButton(toolName)
```

## Parameters

---

toolName	A string identifying the custom tool button.
----------	--

---

## Returns

undefined

## setCurrentTool

Sets the current tool to the one with the specified ID.

## Syntax

```
setCurrentTool(toolName)
```

## Parameters

---

toolName	A string identifying the tool.
----------	--------------------------------

---

## Returns

undefined

## setCustomMenuItemChecked

Acrobat 7.0.7

Sets the checked state of the provided custom menu item.



## Syntax

```
setCustomMenuItemChecked( menuItemName, checkedState )
```

## Parameters

menuItemName	A string identifying the name of the custom menu item.
checkedState	A Boolean value determining whether the menu should be checked.

## Returns

undefined

## setView

Acrobat 9.0.

Sets the current view for the annotation.

See the related method, [getView](#), for getting the view by its index.

## Syntax

```
setView( index, animate)
```

## Parameters

index	The integer index of the view to be set .
animate	(Optional) A Boolean value, when <code>true</code> , indicates that the view should be animated to when set.

## Returns

undefined

## setView

Acrobat 9.0

Sets the current view for the annotation.

See the related method, [getView](#), for getting the view by its name.

## Syntax

```
setView( name, animate)
```

## Parameters

---

<code>menuItemName</code>	The string name of the view to be set.
<code>checkedState</code>	(Optional) A Boolean value, when <code>true</code> , indicates that the view should be animated to when set.

---

## Returns

undefined

## Scene

An object that represents the hierarchy of the 3D related content, including [Animation](#), [Light](#), [Material](#), and [Mesh](#) objects. The variable `scene` is a global reference to this object.

Related objects are [Animation](#), [Light](#), [Material](#) and [Mesh](#).

### Properties

Property	Type	Access	Description
<code>ambientIlluminationColor</code>	Color	R	Modulates the ambient Color of all materials.
<code>animations</code>	SceneObjectList	R	A list of all <a href="#">Animation</a> objects.
<code>cameras</code>	SceneObjectList	R	A list of all <a href="#">Camera</a> objects in the <a href="#">Scene</a> .
<code>defaultRenderOptions</code>	RenderOptions	R	A set of all default rendering options for the <a href="#">Scene</a> .
<code>gridMode</code>	string	R/W	Acrobat 7.0.7 The display style of the grid that represents a portion of the ground plane in the <a href="#">Scene</a> . It can have one of the following values: <ul style="list-style-type: none"><li>• "off" (no grid)</li><li>• "wire" (a wireframe grid)</li><li>• "solid" (a solid checkerboard grid)</li><li>• "transparent" (a semi-transparent checkerboard grid)</li></ul>
<code>GRID_MODE_OFF</code>	string	R	Acrobat 7.0.7 A string constant for the grid mode of "off".
<code>GRID_MODE_SOLID</code>	string	R	Acrobat 7.0.7 A string constant for the grid mode of "solid".
<code>GRID_MODE_TRANSPARENT</code>	string	R	Acrobat 7.0.7 A string constant for the grid mode of "transparent".

Property	Type	Access	Description
GRID_MODE_WIRE	string	R	Acrobat 7.0.7 A string constant for the grid mode of "wire".
gridSize	number	R	Acrobat 7.0.7 The number of squares on the ground plane grid.
lengthUnits	number	R	The scale of a unit of length, specified in meters.
LIGHT_MODE_FILE	string	R	Acrobat 7.0.7 A string constant for the light mode of "file".
LIGHT_MODE_NONE	string	R	Acrobat 7.0.7 A string constant for the light mode of "none".
LIGHT_MODE_WHITE	string	R	Acrobat 7.0.7 A string constant for the light mode of "white".
LIGHT_MODE_DAY	string	R	Acrobat 7.0.7 A string constant for the light mode of "day".
LIGHT_MODE_BRIGHT	string	R	Acrobat 7.0.7 A string constant for the light mode of "bright".
LIGHT_MODE_RGB	string	R	Acrobat 7.0.7 A string constant for the light mode of "rgb".
LIGHT_MODE_NIGHT	string	R	Acrobat 7.0.7 A string constant for the light mode of "night".
LIGHT_MODE_BLUE	string	R	Acrobat 7.0.7 A string constant for the light mode of "blue".
LIGHT_MODE_RED	string	R	Acrobat 7.0.7 A string constant for the light mode of "red".

Property	Type	Access	Description
LIGHT_MODE_CUBE	string	R	Acrobat 7.0.7 A string constant for the light mode of "cube".
LIGHT_MODE_CAD	string	R	Acrobat 7.0.7 A string constant for the light mode of "cad".
LIGHT_MODE_HEADLAMP	string	R	Acrobat 7.0.7 A string constant for the light mode of "headlamp".
lights	SceneObjectList	R	A list of all <code>Light</code> objects in the <code>Scene</code> .
lightScaleFactor	number	R/W	A uniform scale factor for all <code>Light</code> objects in the <code>Scene</code> .
lightScheme	string	R/W	Acrobat 7.0.7 The current, preconfigured lighting scheme for the <code>Scene</code> . It can take one of the following values: <ul style="list-style-type: none"> <li>• "file"</li> <li>• "none"</li> <li>• "white"</li> <li>• "day"</li> <li>• "bright"</li> <li>• "rgb"</li> <li>• "night"</li> <li>• "blue"</li> <li>• "red"</li> <li>• "cube"</li> <li>• "cad"</li> <li>• "headlamp"</li> </ul>
materials	SceneObjectList	R	A list of all <code>Material</code> objects.
meshes	SceneObjectList	R	A list of all <code>Mesh</code> objects in the <code>Scene</code> .

Property	Type	Access	Description
nodes	SceneObjectList	R	A list of all Node objects except the default Camera and default Light objects.
outlineAngle	number	R/W	Acrobat 7.0.7 The crease angle (in degrees) for the appearance of lines in Illustration render modes.
showGrid	Boolean	R/W	Acrobat 7.0.7 Determines whether the ground plane grid is displayed.
renderDoubleSided	Boolean	R/W	Acrobat 8.1 Toggles if backfacing polygons are rendered.
renderMode	string	R/W	Acrobat 7.0.7 The default rendering type for all objects in the Scene, which can be one of the following values: <ul style="list-style-type: none"> <li>• "default"</li> <li>• "bounding box"</li> <li>• "transparent bounding box"</li> <li>• "transparent bounding box outline"</li> <li>• "vertices"</li> <li>• "shaded vertices"</li> <li>• "wireframe"</li> <li>• "shaded wireframe"</li> <li>• "solid"</li> <li>• "transparent"</li> <li>• "solid wireframe"</li> <li>• "transparent wireframe"</li> <li>• "illustration"</li> <li>• "solid outline"</li> <li>• "shaded illustration"</li> <li>• "hidden wireframe"</li> </ul>

Property	Type	Access	Description
RENDER_MODE_DEFAULT	string	R	Acrobat 7.0.7 A string constant for the render mode of "default".
RENDER_MODE_BOUNDING_BOX	string	R	Acrobat 7.0.7 A string constant for the render mode of "bounding box".
RENDER_MODE_TRANSPARENT_BOUNDING_BOX	string	R	Acrobat 7.0.7 A string constant for the render mode of "transparent bounding box".
RENDER_MODE_TRANSPARENT_BOUNDING_BOX_OUTLINE	string	R	Acrobat 7.0.7 A string constant for the render mode of "transparent bounding box outline".
RENDER_MODE_VERTICES	string	R	Acrobat 7.0.7 A string constant for the render mode of "vertices".
RENDER_MODE_SHADED_VERTICES	string	R	Acrobat 7.0.7 A string constant for the render mode of "shaded vertices".
RENDER_MODE_WIREFRAME	string	R	Acrobat 7.0.7 A string constant for the render mode of "wireframe".
RENDER_MODE_SHADED_WIREFRAME	string	R	Acrobat 7.0.7 A string constant for the render mode of "shaded wireframe".
RENDER_MODE_SOLID	string	R	Acrobat 7.0.7 A string constant for the render mode of "solid".

Property	Type	Access	Description
RENDER_MODE_TRANSPARENT	string	R	Acrobat 7.0.7 A string constant for the render mode of "transparent".
RENDER_MODE_SOLID_WIREFRAME	string	R	Acrobat 7.0.7 A string constant for the render mode of "solid wireframe".
RENDER_MODE_TRANSPARENT_WIREFRAME	string	R	Acrobat 7.0.7 A string constant for the render mode of "transparent wireframe".
RENDER_MODE_ILLUSTRATION	string	R	Acrobat 7.0.7 A string constant for the render mode of "illustration".
RENDER_MODE_SOLID_OUTLINE	string	R	Acrobat 7.0.7 A string constant for the render mode of "solid outline".
RENDER_MODE_SHADED_ILLUSTRATION	string	R	Acrobat 7.0.7 A string constant for the render mode of "shaded illustration".
RENDER_MODE_HIDDEN_WIREFRAME	string	R	Acrobat 7.0.7 A string constant for the render mode of "hidden wireframe".
selectedNode	Node	R/W	Acrobat 8.1 The currently selected Node.
showAxes	Boolean	R/W	Acrobat 7.0.7 Determines whether the world axes are displayed.
showOrientationAxes	Boolean	R/W	Acrobat 9.0 Determines whether the orientation axes are displayed.



Property	Type	Access	Description
smoothing	Boolean	R/W	Acrobat 7.0.7 When <code>true</code> , smoothing is enabled for meshes in the scene.
smoothingAngle	number	R/W	Acrobat 7.0.7 The default smoothing angle (in degrees) for meshes in the scene.
smoothingOverride	Boolean	R/W	Acrobat 7.0.7 When set to <code>true</code> , overrides the smoothing values from the loaded model file.

## activateAnimation

Sets the given `Animation` to be active on its `Node` objects.

### Syntax

```
activateAnimation(animation)
```

### Parameters

<code>animation</code>	The <code>Animation</code> object to be activated.
------------------------	--

### Returns

undefined

## addFlashForeground

Acrobat 9.0

Adds the provided `FlashMovie` as a foreground element within the 3D scene.

### Syntax

```
addFlashForeground(movie)
```

### Parameters

<code>movie</code>	The <code>FlashMovie</code> to be added as a foreground element.
--------------------	--

## Returns

undefined

## addModel

Adds a model Resource to the top level of the Scene.

## Syntax

```
addModel (modelRes)
```

## Parameters

modelRes	The Resource object to be added.
----------	----------------------------------

## Returns

A Node object representing the top-level Mesh of the loaded model.

## createClippingPlane

Creates a new clipping plane.

## Syntax

```
createClippingPlane ()
```

## Returns

A ClippingPlane object.

## createLight

Creates a new Light and attaches it to the Scene .

## Syntax

```
createLight ()
```

## Returns

A Light object.

## createSquareMesh

Creates a Mesh that is a unit square. The default UV parameterization fits once in U and V.

## Syntax

```
createSquareMesh (sizeX, sizeY, name)
```

## Parameters

sizeX	Model units in the x direction used to size the Mesh.
sizeY	Model units in the y direction used to size the Mesh.
name	(Optional) The name that is assigned to the Mesh.

## Returns

A Mesh object.

## computeBoundingBox

Computes the BoundingBox of the Scene .

## Syntax

```
computeBoundingBox ( )
```

## Returns

A BoundingBox object.

## update

Applies all changes to the Scene.

## Syntax

```
update ( )
```

## Returns

undefined

# SceneObject

The base type for objects within the Scene, including Animation, Material, and Node objects.

Related objects are [Scene](#), [Animation](#), [Light](#), [Material](#), and [Mesh](#).

## Properties

Property	Type	Description
name	string	The name of the SceneObject object.
objectGUID	string	Deprecated A value that uniquely identifies the SceneObject in custom data. This property has a default value.
objectID	integer	An unsigned 32-bit value that uniquely identifies the SceneObject. This property can be assigned, but it does not have a default value. It always returns 0.

# SceneObjectList

A structure that contains references to several `SceneObject` objects.

## Properties

Property	Type	Access	Description
count	integer	R	The number of elements in the <code>SceneObjectList</code> .

## getByGUID

Deprecated

Obtains the specified `SceneObject` object from the list.

## Syntax

`getByGUID(guid)`

## Parameters

guid	A string representing the GUID for the specified element.
------	---

## Returns

A `SceneObject` object.

## getByID

Obtains the specified `SceneObject` object from the list

## Syntax

`getByID(id)`

## Parameters

id	An integer representing the object identifier for the specified <code>SceneObject</code> object.
----	--

## Returns

A `SceneObject` object.

## getByIndex

Obtains the specified `SceneObject` object from the list.

## Syntax

`getByIndex (index)`

## Parameters

index	An integer representing the index of the specified <code>SceneObject</code> object.
-------	---

## Returns

A `SceneObject` object.

## getByName

Obtains the specified `SceneObject` object from the list.

## Syntax

`getByName (name)`

## Parameters

name	A string representing the name of the specified <code>SceneObject</code> object.
------	--

## Returns

A `SceneObject` object.

## removeAll

Deprecated

Removes all the `SceneObject` objects from the list, but does not delete them from the `Scene`.

## Syntax

`removeAll ()`

## Returns

undefined

## removeByIndex

Deprecated

Removes the specified `SceneObject` object from the list, but does not delete it from the `Scene`.

## Syntax

```
removeByIndex (index)
```

## Parameters

index	An index to the specified element.
-------	------------------------------------

## Returns

undefined

## removeItem

Deprecated

Removes a `SceneObject` object from the list, but does not delete it from the `Scene`.

## Syntax

```
removeItem (scene)
```

## Parameters

scene	A scene object that is to be removed.
-------	---------------------------------------

## Returns

undefined

## ScrollWheelEvent

(Acrobat 8.1) An object that is passed as an argument to the `onEvent` method of the [ScrollWheelEventHandler](#) object. A `ScrollWheelEvent` object is created when the mouse scroll wheel is activated over an active 3D Canvas object.

### Properties

Property	Type	Access	Description
<code>canvas</code>	Canvas	R	The Canvas in which the <code>ScrollWheelEvent</code> took place.
<code>canvasPixelHeight</code>	integer	R	The height, measured in pixels, of the Canvas in which the <code>ScrollWheelEvent</code> took place.
<code>canvasPixelWidth</code>	integer	R	The width, measured in pixels, of the Canvas in which the <code>ScrollWheelEvent</code> took place.
<code>ctrlKeyDown</code>	Boolean	R	Determines whether the Ctrl key (Windows) or Command key (Mac OS) was pressed.
<code>currentTool</code>	string	R	The name of the current tool.
<code>deltaY</code>	number	R	The amount the scroll wheel was moved in the Y direction.
<code>shiftKeyDown</code>	Boolean	R	Determines whether the Shift key was pressed.



# ScrollWheelEventHandler

(Acrobat 8.1) An object that exposes a callback mechanism that allows a function to be evaluated when an event occurs. Event handlers are registered with the `Runtime` method [addEventHandler](#).

## ScrollWheelEventHandler

A constructor that creates a new `ScrollWheelEventHandler`.

### Syntax

```
new ScrollWheelEventHandler()
```

### Returns

A `ScrollWheelEventHandler` object.

## onEvent

A method that is called when the scroll wheel is used in an active 3D annotation.

### Syntax

```
onEvent(event)
```

### Parameters

event	A <code>ScrollWheelEvent</code> object representing the event.
-------	--

### Returns

undefined

## SelectionEvent

(Acrobat 8.1) An object that is passed as an argument to the `onEvent` method of the [SelectionEventHandler](#) object.

A `SelectionEvent` object is created when an object is selected from an active 3D Canvas object or from a model tree. If the selection is made from a Canvas object, a `MouseEvent` is also created.

### Properties

Property	Type	Access	Description
<code>node</code>	Node	R	The Node that is the target of the selection change.
<code>selected</code>	Boolean	R	The selected state of the target Node.

# SelectionEventHandler

(Acrobat 8.1) An object that exposes a callback mechanism that allows a function to be evaluated when an event occurs. Event handlers are registered with the `Runtime` method `addEventHandler`.

## SelectionEventHandler

A constructor that creates a new `SelectionEventHandler` object.

### Syntax

```
new SelectionEventHandler()
```

### Returns

A `SelectionEventHandler` object.

## onEvent

A method that is called when the selection state changes in an active 3D annotation.

### Syntax

```
onEvent (event)
```

### Parameters

---

event	A <code>ScrollWheelEvent</code> object representing the event.
-------	--

---

### Returns

undefined

## StateEvent

Acrobat 9.0

An object that is passed as an argument to the `onEvent` method of the `StateEventHandler` object. A `StateEvent` object is created when state data must be stored or loaded for the scene, such as when a new comment view is created or invoked for the annotation.

### Properties

Property	Type	Access	Description
<code>stateString</code>	string	R	If the <code>SaveEvent</code> type is "load", this property contains the state data that was stored as part of the corresponding "save" <code>StateEvent</code> . If the <code>SaveEvent</code> type is "save", the <code>stateString</code> is undefined.
<code>type</code>	string	R	The type of <code>StateEvent</code> , this property has a value of either "load" or "save".
<code>TYPE_LOAD</code>	string	R	A string constant for the <code>StateEvent</code> type of "load".  The state data that was stored as part of the original <code>stateEvent</code> .
<code>TYPE_SAVE</code>	string	R	A string constant for the <code>StateEvent</code> type of "save".

# StateEventHandler

Acrobat 9.0

An object that exposes a callback mechanism that allows a function to be evaluated when a state event occurs. Event handlers are registered with the `Runtime` method `addEventHandler`.

## onEvent

A method that is called when state data must be stored or loaded for the annotation. The return value is stored as the `stateString` for the given `StateEvent`.

## Syntax

```
onEvent (event)
```

## Parameters

event	A <code>StateEvent</code> object representing the event.
-------	--

## Returns

string or undefined

## StateEventHandler

The constructor that creates a new `StateEventHandler`.

## Syntax

```
new StateEventHandler()
```

## Returns

A `StateEventHandler` object.

# Texture

A `Texture` object represents the mapping of a texture. All `Texture` properties have read-write permissions.

## Properties

Property	Type	Description
<code>amount</code>	number	The degree to which the <code>Texture</code> is applied, which can be a value from 0.0 to 1.0.
<code>angle</code>	number	The rotation angle, measured in degrees, of the map.
<code>clampU</code>	Boolean	Determines whether the map should be clamped in the U direction.
<code>clampV</code>	Boolean	Determines whether the map should be clamped in the V direction.
<code>image</code>	Image	Acrobat 7.0.7 The <code>Image</code> to be used by the <code>Texture</code> .
<code>modulate</code>	Boolean	Determines whether to set the blend mode of the <code>Texture</code> with its corresponding color.
<code>uOffset</code>	number	The U-offset of the given map.
<code>uScale</code>	number	The U-scale of the given map.
<code>use3DSStyleMapping</code>	Boolean	Determines whether to use 3D Studio style map parameterizations.
<code>vOffset</code>	number	The V-offset of the given map.
<code>vScale</code>	number	The V-scale of the given map.

## getImage

Deprecated

Gets the `Image` currently used by the `Texture`.

## Syntax

```
getImage ()
```

## Returns

The `Image` currently being used.

## setImage

Deprecated

Sets the Image to be used by the Texture.

### Syntax

```
setImage (image)
```

### Parameters

---

image	The Image to be used.
-------	-----------------------

---

### Returns

undefined

## TimeEvent

An object that is passed as an argument to the `TimeEventHandler` object's `onEvent` method.

### Properties

Property	Type	Access	Description
<code>deltaTime</code>	number	R	The difference between the current time and the last time.
<code>time</code>	number	R	The current time.



# TimeEventHandler

An object that exposes a callback mechanism that allows a function to be evaluated when an event occurs. Event handlers are registered with the `Runtime.addEventHandler` method.

## TimeEventHandler

A constructor that creates a new `TimeEventHandler` object.

### Syntax

```
new TimeEventHandler()
```

### Returns

A `TimeEventHandler` object.

## onEvent

A method that is called when simulation time is incremented in an active 3D annotation.

### Syntax

```
onEvent(event)
```

### Parameters

---

event	A <code>TimeEvent</code> object representing the event.
-------	---

---

### Returns

undefined

# ToolEvent

An object that is passed as an argument to the `onEvent` method of the [ToolEventHandler](#) object.

## Properties

Property	Type	Access	Description
<code>canvas</code>	Canvas	R	The Canvas that is the target of the ToolEvent.
<code>canvasPixelHeight</code>	integer	R	The height, measured in pixels, of the Canvas for which the ToolEvent is intended.
<code>canvasPixelWidth</code>	integer	R	The width, measured in pixels, of the Canvas for which the ToolEvent is intended.
<code>currentTool</code>	string	R	The name of the current tool.
<code>toolName</code>	string	R	The name of the tool that was clicked.

# ToolEventHandler

An object that exposes a callback mechanism that allows a function to be evaluated when an event occurs. Event handlers are registered with the Runtime [addEventListener](#) method.

## ToolEventHandler

A constructor that creates a new ToolEventHandler object.

### Syntax

```
new ToolEventHandler ()
```

### Returns

A ToolEventHandler object.

## onEvent

A method that is called when a tool button is pressed on the 3D toolbar.

### Syntax

```
onEvent (event)
```

### Parameters

---

event	A ToolEvent object representing the event.
-------	--

---

### Returns

undefined

## Vector3

An object comprised of three values that represent a point in space or a direction and magnitude.

### Properties

Property	Type	Access	Description
x	number	R/W	The x component of the <code>Vector3</code> object.
y	number	R/W	The y component of the <code>Vector3</code> object.
z	number	R/W	The z component of the <code>Vector3</code> object.
length	number	R	The length of the <code>Vector3</code> object.

## Vector3

A constructor that initializes the new object to (0.0, 0.0, 0.0).

### Syntax

```
new Vector3()
```

### Returns

A `Vector3` object.

## Vector3

A constructor that initializes the new object to the specified components.

### Syntax

```
new Vector3(x, y, z)
```

### Parameters

x	The x component used to initialize the new object.
y	The y component used to initialize the new object.
z	The z component used to initialize the new object.

### Returns

A `Vector3` object.

## add

Adds the specified `Vector3` to the current one.

### Syntax

```
add(offset)
```

### Parameters

<code>offset</code>	The <code>Vector3</code> object to be added to the current one.
---------------------	---

### Returns

A `Vector3` object.

## addInPlace

Adds the specified `Vector3` to the current one, and updates the current `Vector3` with the resulting value.

### Syntax

```
addInPlace(offset)
```

### Parameters

<code>offset</code>	The <code>Vector3</code> object to be added to the current one.
---------------------	---

### Returns

undefined

## addScaled

Adds the specified `Vector3` with the scaled offset to the current one.

### Syntax

```
addScaled(offset, scale)
```

### Parameters

<code>offset</code>	The <code>Vector3</code> object to be added to the current one.
<code>scale</code>	The scaling factor for the <code>offset</code> .

## Returns

A `Vector3` object.

## addScaledInPlace

Adds the specified `Vector3` with the scaled offset to the current one, and updates the current `Vector3` with the resulting value.

## Syntax

```
addScaledInPlace(offset, scale)
```

## Parameters

<code>offset</code>	The <code>Vector3</code> object to be added to the current one.
<code>scale</code>	The scaling factor for the <code>offset</code> .

## Returns

undefined

## blend

Blends the current and specified `Vector3` by the specified degree.

## Syntax

```
blend(vec, blendFactor)
```

## Parameters

<code>vec</code>	The <code>Vector3</code> object to be blended with the current one.
<code>blendFactor</code>	The degree of blending to be applied, which can be a value from 0.0 to 1.0.

## Returns

A `Vector3` object.

## blendInPlace

Blends the current and specified `Vector3` by the specified degree, and updates the current `Vector3` with the resulting value.

## Syntax

```
blendInPlace(vec, blendFactor)
```

## Parameters

<code>vec</code>	The <code>Vector3</code> object to be blended with the current one.
<code>blendFactor</code>	The degree of blending to be applied, which can be a value from 0.0 to 1.0.

## Returns

undefined

## cross

Calculates the cross product between the specified `Vector3` and the current one.

## Syntax

`cross (vec)`

## Parameters

<code>vec</code>	The <code>Vector3</code> object to be used in calculating the cross product.
------------------	--

## Returns

A `Vector3` object.

## dot

Calculates the dot product between the specified `Vector3` and the current one.

## Syntax

`dot (vec)`

## Parameters

<code>vec</code>	The <code>Vector3</code> object to be used in calculating the dot product.
------------------	--

## Returns

A number value representing the scalar dot product.

## normalize

Normalizes the current `Vector3`.

## Syntax

```
normalize()
```

## Returns

undefined

## scale

Scales the current `Vector3`.

## Syntax

```
scale(scale)
```

## Parameters

scale	A number value used to scale the current <code>Vector3</code> .
-------	---

## Returns

A `Vector3` object.

## scaleInPlace

Scales the current `Vector3`, and updates the current `Vector3` with the resulting value.

## Syntax

```
scaleInPlace(scale)
```

## Parameters

scale	A number value used to scale the current <code>Vector3</code> .
-------	---

## Returns

undefined

## set

Sets the current `Vector3` to the value contained in the specified `Vector3`.

## Syntax

```
set(vec)
```



## Parameters

<code>vec</code>	The <code>Vector3</code> used to set the current <code>Vector3</code> .
------------------	---

## Returns

undefined

## set

Acrobat 7.0.7

Sets the current `Vector3` to the values contained in the specified components.

## Syntax

```
set (x, y, z)
```

## Parameters

<code>x</code>	The x component used to set the current <code>Vector3</code> .
<code>y</code>	The y component used to set the current <code>Vector3</code> .
<code>z</code>	The z component used to set the current <code>Vector3</code> .

## Returns

undefined

## set3

Deprecated

Sets the current `Vector3` to the values contained in the specified components.

## Syntax

```
set3 (x, y, z)
```

## Parameters

<code>x</code>	The x component used to set the current <code>Vector3</code> .
<code>y</code>	The y component used to set the current <code>Vector3</code> .
<code>z</code>	The z component used to set the current <code>Vector3</code> .

## Returns

undefined

## subtract

Subtracts the specified `Vector3` from the current one.

## Syntax

```
subtract (offset)
```

## Parameters

offset	The <code>Vector3</code> object to be subtracted from the current one.
--------	--

## Returns

A `Vector3` object.

## subtractInPlace

Subtracts the specified `Vector3` from the current one, and updates the current `Vector3` with the resulting value.

## Syntax

```
subtractInPlace (offset)
```

## Parameters

offset	The <code>Vector3</code> object to be subtracted from the current one.
--------	--

## Returns

undefined

# View

Acrobat 9.0

An object that represents a named view for the annotation.

See the `viewCount` property and the [getView](#) and [setView](#) methods of the [Runtime](#) object.

## Properties

Property	Type	Access	Description
name	string	R	The name of the view.

## 3 New Features and Changes

---

This chapter summarizes the new features and changes introduced in Acrobat.

### Acrobat X changes

There are no changes to JavaScript for Acrobat 3D Annotations in Acrobat X.

### Acrobat 9.0 changes

This section describes the changes introduced in Acrobat 9 Pro Extended.

#### New objects

The following new objects are introduced to support Rich Media annotations and Geospatial features in Acrobat 9.0:

- [FlashEvent](#) object: This new object has these properties: `command`, `target`, `type`, `TYPE_COMMAND`, `TYPE_PROGRESS`, `TYPE_STATECHANGE`, and `value`.
- [FlashEventHandler](#) object: This new object has one property, `target`, and these methods: `onEvent` and `FlashEventHandler`.
- [FlashMovie](#) object: This new object has these properties: `alignMode`, `ALIGN_MODE_BOTTOM`, `ALIGN_MODE_LEFT`, `ALIGN_MODE_RIGHT`, `ALIGN_MODE_TOP`, `backgroundColor`, `desiredResolutionX`, `desiredResolutionY`, `frameNum`, `hitEnabled`, `id`, `loop`, `opacity`, `percentLoaded`, `playing`, `quality`, `readyState`, `resolutionType`, `RESOLUTION_TYPE_CUSTOM`, `RESOLUTION_TYPE_MOVIE`, `RESOLUTION_TYPE_WINDOW`, `scaleMode`, `SCALE_MODE_EXACT_FIT`, `SCALE_MODE_NO_BORDER`, `SCALE_MODE_SHOW_ALL`, `totalFrames`, `x`, and `y`. It also has these methods: `FlashMovie`, `call`, `getVariable`, `gotoFrame`, `isPlaying`, `pan`, `play`, `rewind`, `setVariable`, `setZoomRect`, `stop`, and `zoom`.
- [StateEvent](#) object: This new object has these properties: `stateString`, `type`, `TYPE_LOAD`, and `TYPE_SAVE`.
- [StateEventHandler](#) object: This new object has these methods: `onEvent` and `StateEventHandler`.
- [View](#) object: This new object has the `name` property.

#### Additional properties and methods in existing objects

- The [Background](#) object has one additional property, `FlashMovie`.
- The [Resource](#) object has additional properties: `type` (a new value of "flash") and `TYPE_FLASH`.
- This Material object has one additional method, [attachFlashMovie](#).
- The [Runtime](#) object has additional properties: `overrideFlyTool`, `TOOL_NAME_FLY`, and `viewCount`. It also has additional methods: [pause](#), [play](#), [getView](#), and [setView](#).
- The [Scene](#) object has a new property, `showOrientationAxes`, and a new method, [addFlashForeground](#).

## APIs for versions earlier than 9.0

The following properties and methods were available but undocumented in earlier versions of this reference.

- Two properties were defined for the [Camera](#) object in version 7, `absoluteBindingScale` and `useAbsoluteBinding`.
- Two `set` methods were defined for the [Matrix4x4](#) object in version 8.1, one takes an array argument and the other a list of numbers.
- The `setCustomMenuItemChecked` method was defined for the `Runtime` object in version 7.0.7.
- The `opacity` property was defined for the [Node](#) object in version 7.0.7.

## API changes

- The `computeBoundingBox` method is documented as a method of the `Mesh` object, not the `Node` object, as previously published.
- The property previously documented for the [Runtime](#) object as `overrideWheelSpeed` was incorrect. It is now properly identified as `overrideScrollWheel`, and the description changed.
- The `RENDER_MODE_SOLID_WIREFRAME` property for the [Scene](#) object was incorrectly documented as `RENDER_MODE_SHADED_SOLID_WIREFRAME`.
- Deprecate the methods `removeAll`, `removeByIndex`, and `removeItem` for the [SceneObjectList](#) object. Their descriptions were modified to indicate that these methods remove elements from the `SceneObjectList`, but not from the `Scene`.

## Acrobat 8.1 changes

This section describes the changes introduced in Acrobat 8.1.

### New objects

The following objects are new: [ScrollWheelEvent](#), [ScrollWheelEventHandler](#), [SelectionEvent](#), and [SelectionEventHandler](#).

### Additional properties in existing objects

The [HitInfo](#) object has additional properties: `material`, `surfaceNormal`, and `textureCoordinate`.

The [Node](#) object has an additional property: `metadataString`.

The [Light](#) object has an additional property: `directionLocal` (Acrobat 7, but previously undocumented).

The [Runtime](#) object has additional properties: `canvasCount`, `overrideSpinTool`, `scrollWheelSpeed`, `speedThreshold`, `strafeSpeed`, and `walkSpeed`.

The [Scene](#) object has additional properties: `node` and `selected`.

## Deprecated objects or properties

The following APIs were deprecated:

[CamaraEvent](#).isNewCanvas (a property)

[Dummy](#) (an object)

[Procedural](#) (an object)

[SceneObject](#).objectGUID (a property)

[SceneObject](#).getByGUID (a method)

## Acrobat 8.0 changes

This section describes the changes introduced in Acrobat 8.0.

### Additional properties in existing objects

The [Runtime](#) object has additional properties: `overrideSpinTool` and `TOOL_NAME_SPIN`.

# Index

## Numbers

- 3D JavaScript engine 9
  - accessing 9
  - accessing using the SceneContext3D object 9

## A

- absoluteBindingScale property 21
- accessing the 3D JavaScript engine 9
  - using the SceneContext3d object 9
- activateAnimation method 97
- add method 117
- addCustomMenuItem method 83
- addCustomToolButton method 83
- addEventHandler method 84
- addFlashForeground method 97
- addInPlace method 117
- addModel method 98
- addScaled method 117
- addScaledInPlace method 118
- ALIGN\_MODE\_BOTTOM property 35
- ALIGN\_MODE\_LEFT property 35
- ALIGN\_MODE\_RIGHT property 35
- alignMode property 35
- ambientColor property 49
- ambientIlluminationColor property 91
- ambientTexture property 49
- amount property 110
- angle property 110
- Animation object
  - about 16
  - currentTime property 16
  - endTime property 16
  - framesPerSecond property 16
  - length property 16
  - startTime property 16
- animations property 91
- Annot3D.context3D property 9
- ATTENUATION\_ABC property 47
- ATTENUATION\_NONE property 47
- attenuationA property 47
- attenuationB property 47
- attenuationC property 47
- attenuationType property 47

## B

- Background object
  - about 17
  - FlashMovie property 17
  - getColor method 17
  - getImage method 17
  - image property 17
  - setColor method 17

- setImage method 18
- background property 26
- binding property 21, 24
- BINDING\_HORIZONTAL property 21
- BINDING\_MAX property 21
- BINDING\_MIN property 21
- BINDING\_VERTICAL property 21
- blend method 118
- blendInPlace method 118
- Bone object 19
- BoundingBox object
  - about 20
  - center property 20
  - max property 20
  - min property 20
- boundingBoxColor property 77
- brightness property 47
- bumpTexture property 49
- BUTTON\_TYPE\_PUSH property 80
- BUTTON\_TYPE\_TOOL property 80

## C

- call method 37
- Camera object
  - about 21
  - absoluteBindingScale property 21
  - binding property 21
  - BINDING\_HORIZONTAL property 21
  - BINDING\_MAX property 21
  - BINDING\_MIN property 21
  - BINDING\_VERTICAL property 21
  - far property 21
  - fov property 21
  - getDirectionFromScreen method 23
  - getScreenFromPosition method 22
  - near property 21
  - position property 21
  - positionLocal property 21
  - projectionType property 22
  - roll property 22
  - target property 22
  - targetPosition property 22
  - targetPositionLocal property 22
  - TYPE\_ORTHOGRAPHIC property 22
  - TYPE\_PERSPECTIVE property 22
  - up property 22
  - upLocal property 22
  - viewPlaneSize property 22
- CameraEvent object
  - about 24
  - binding property 24
  - canvas property 24
  - currentTool property 24

- far property 24
- fov property 24
- isNewCanvas property 24
- near property 24
- projectionType property 24
- targetDistance property 24
- transform property 24
- viewPlaneSize property 24
- CameraEventHandler method 25
- CameraEventHandler object
  - about 25
  - CameraEventHandler method 25
- cameras property 91
- Canvas object
  - about 26
  - background property 26
  - getCamera method 26
  - setCamera method 26
- canvas property 24, 44, 62, 65, 75, 104, 114
- canvasCount property 80
- canvasPixelHeight property 44, 65, 75, 104, 114
- canvasPixelWidth property 44, 65, 75, 104, 114
- center property 20
- characterCode property 44
- clampU property 110
- clampV property 110
- ClippingPlane object
  - about 27
  - remove method 27
- clippingPlaneColor property 77
- clippingPlaneIntersectionColor property 77
- Color method 28
- Color object
  - about 28
  - Color method 28
  - r, g, b properties 28
  - set method 28
  - set3 method 29
- color property 47
- command property 33
- computeBoundingBox method 64, 99
- Console object
  - about 31
  - print method 31
  - println method 31
- corrections made in Acrobat 9.0 125
- count property 101
- createClippingPlane method 98
- createLight method 98
- createSquareMesh method 98
- cross method 119
- ctrlKeyDown property 45, 65, 80, 104
- currentTime property 16
- currentTool property 24, 45, 62, 65, 75, 104, 114

## D

- defaultAmbientColor property 77
- defaultDiffuseColor property 77
- defaultEmissiveColor property 77

- defaultRenderOptions property 91
- defaultSpecularColor property 77
- deltaTime property 112
- deltaY property 104
- desiredResolutionY property 35
- detachFromCurrentAnimation method 70
- determinant property 51
- diffuseColor property 49
- diffuseTexture property 49
- direction property 48
- directionLocal property 48
- disableTool method 84
- distance property 42
- dot method 119
- Dummy object 32

## E

- emissiveColor property 49
- emissiveTexture property 49
- enableTool method 84
- endTime property 16
- event handlers
  - types 12
- eventHandlerCount property 80

## F

- far property 21, 24
- firstChild property 69
- FlashEventHandler method 34
- FlashEventHandler object
  - about 34
  - FlashEventHandler method 34
  - onEvent method 34
  - target property 34
- FlashMovie method 37
- FlashMovie object
  - about 35
  - ALIGN\_MODE\_BOTTOM property 35
  - ALIGN\_MODE\_LEFT property 35
  - ALIGN\_MODE\_RIGHT property 35
  - alignMode Property 35
  - call method 37
  - desiredResolutionY property 35
  - FlashMovie method 37
  - frameNum property 35
  - getVariable method 38
  - gotoFrame method 38
  - hitEnabled property 35
  - id property 36
  - isPlaying method 39
  - loop property 36
  - opacity property 36
  - pan method 39
  - percentLoaded property 36
  - play method 39
  - playing property 36
  - quality property 36
  - readyState property 36
  - resolution Type property 36



- RESOLUTION\_TYPE\_CUSTOM property 36
- RESOLUTION\_TYPE\_MOVIE property 36
- RESOLUTION\_TYPE\_WINDOW property 36
- rewind method 39
- SCALE\_MODE\_EXACT\_FIT property 36
- SCALE\_MODE\_NO\_BORDER property 36
- SCALE\_MODE\_SHOW\_ALL property 37
- scaleMode property 36
- setVariable method 40
- setZoomRect 40
- stop method 41
- totalFrames property 37
- x property 37
- y property 37
- zoom method 41
- FlashMovie property 17
- Flashovers object
  - about 33
  - command property 33
  - target property 33
  - type property 33
  - TYPE\_COMMAND property 33
  - TYPE\_PROGRESS property 33
  - TYPE\_STATECHANGE property 33
  - value property 33
- fov property 21, 24
- frameNum property 35
- framesPerSecond property 16

## G

- getByGUID method 101
- getByID method 101
- getByIndex method 101
- getByName method 102
- getCamera method 26
- getColor method 17
- getDirectionFromScreen method 23
- getEventHandler method 85
- getImage method 17, 110
- getRendererName method 85
- getScreenFromPosition method 22
- getVariable method 38
- getView method 85, 86
- gotoFrame method 38
- GRID\_MODE\_OFF property 91
- GRID\_MODE\_SOLID property 91
- GRID\_MODE\_TRANSPARENT property 91
- GRID\_MODE\_WIRE property 92
- gridMode property 91
- gridSize property 92

## H

- height property 43
- hitEnabled property 35, 69
- HitInfo object
  - about 42
  - distance property 42
  - material property 42
  - position property 42

- surfaceNormal property 42
- target property 42
- textureCoordinate property 42
- hits property 65
- Host object 42

## I

- id property 36
- illustrationRenderModeFaceColor property 77
- illustrationRenderModeLineColor property 77
- Image method 43
- Image object
  - about 43
  - height property 43
  - Image method 43
  - width property 43
- image property 17, 110
- info property 69
- innerConeAngle property 48
- innerRadius property 48
- instances property 80
- interpolate method 73
- interpolateInPlace method 73
- inverse property 51
- invertInPlace method 52
- isDoubleClick property 65
- isEqual method 52
- isMouseDown property 65
- isMouseHit property 65
- isMouseMove property 65
- isMouseOut property 65
- isMouseOver property 65
- isMouseUp property 65
- isNewCanvas property 24
- isPlaying method 39

## K

- KeyEvent object
  - about 44
  - canvas property 44
  - canvasPixelHeight property 44
  - canvasPixelWidth property 44
  - characterCode property 44
  - ctrlKeyDown property 45
  - currentTool property 45
  - shiftKeyDown property 45
- KeyEventHandler method 46
- KeyEventHandler object
  - about 46
  - KeyEventHandler method 46
  - onEvent method 46

## L

- leftButtonDown property 65
- length property 16, 116
- lengthUnits property 92
- Light object
  - about 47

- ATTENUATION\_ABC property 47
- ATTENUATION\_NONE property 47
- attenuationA property 47
- attenuationB property 47
- attenuationC property 47
- attenuationType property 47
- brightness property 47
- color property 47
- direction property 48
- innerConeAngle property 48
- innerRadius property 48
- outerConeAngle property 48
- outerRadius property 48
- position property 48
- positionLocal property 48
- type property 48
- TYPE\_INFINITE property 48
- TYPE\_POINT property 48
- TYPE\_SPOT property 48
- LIGHT\_MODE\_BLUE property 92
- LIGHT\_MODE\_BRIGHT property 92
- LIGHT\_MODE\_CAD property 93
- LIGHT\_MODE\_CUBE property 93
- LIGHT\_MODE\_DAY property 92
- LIGHT\_MODE\_FILE property 92
- LIGHT\_MODE\_HEADLAMP property 93
- LIGHT\_MODE\_NIGHT property 92
- LIGHT\_MODE\_NONE property 92
- LIGHT\_MODE\_RED property 92
- LIGHT\_MODE\_RGB property 92
- LIGHT\_MODE\_WHITE property 92
- lights property 93
- lightScaleFactor property 93
- lightScheme property 93
- loop property 36

## M

### Material object

- about 49
- ambientColor property 49
- ambientTexture property 49
- bumpTexture property 49
- diffuseColor property 49
- diffuseTexture property 49
- emissiveColor property 49
- emissiveTexture property 49
- opacity property 49
- opacityTexture property 49
- phongExponent property 49
- reflectionStrength property 49
- reflectionTexture property 49
- specularColor property 50
- specularStrength property 50
- material property 42, 64
- materials property 93
- Matrix4x4 method 51
- Matrix4x4 object
  - about 51
  - determinant property 51

- inverse property 51
- invertInPlace method 52
- isEqual method 52
- Matrix4x4 method 51
- multiply method 52
- multiplyInPlace method 52
- rotateAboutLine method 53
- rotateAboutLineInPlace method 54
- rotateAboutVector method 55
- rotateAboutVectorInPlace method 55
- rotateAboutX method 54
- rotateAboutXInPlace method 55
- rotateAboutY method 56
- rotateAboutYInPlace method 56
- rotateAboutZ method 56
- rotateAboutZInPlace method 57
- rotateWithQuaternion method 53
- rotateWithQuaternionInPlace method 53
- scale method 57
- scaleComponent property 51
- scaleInPlace method 58
- set method 58, 59
- setIdentity method 59
- setView method 59
- transformDirection method 60
- transformPosition method 60
- translate method 60
- translateInPlace method 61
- translation property 51
- transpose property 51
- transposeInPlace method 61
- max property 20
- MENU\_ITEM\_TYPE\_CHECKED property 80
- MENU\_ITEM\_TYPE\_DEFAULT property 80
- MenuEvent object
  - about 62
  - canvas property 62
  - currentTool property 62
  - menuItemChecked property 62
  - menuItemName property 62
- MenuEventHandler method 63
- MenuEventHandler object
  - about 63
  - MenuEventHandler method 63
  - onEvent method 63
- menuItemChecked property 62
- menuItemName property 62
- Mesh object
  - about 64
  - computeBoundingBox method 64
  - material property 64
  - renderMode property 64
- meshes property 93
- metadataString property 69
- methods
  - corrections in Acrobat 9.0 125
  - new in Acrobat 9.0 124
  - undocumented in earlier versions 125
- min property 20
- modulate property 110

## MouseEvent object

- about 65
- canvas property 65, 104
- canvasPixelHeight property 65, 104
- canvasPixelWidth property 65, 104
- ctrlKeyDown property 65, 104
- currentTool property 65, 104
- deltaY property 104
- hits property 65
- isDoubleClick property 65
- isMouseDown property 65
- isMouseHit property 65
- isMouseMove property 65
- isMouseOut property 65
- isMouseOver property 65
- isMouseUp property 65
- leftButtonDown property 65
- mouseX property 65
- mouseY property 65
- rightButtonDown property 66
- shiftKeyDown property 66, 104

## MouseEventHandler method 68

## MouseEventHandler object

- about 67
- MouseEventHandler method 68
- onEvent method 68
- onMouseDown property 67
- onMouseHit property 67
- onMouseMove property 67
- onMouseOut property 67
- onMouseOver property 67
- onMouseUp property 67
- reportAllTargets property 68
- target property 68

mouseX property 65

mouseY property 65

multiply method 52

multiplyInPlace method 52

## N

name property 100

near property 21, 24

new methods

- Acrobat 9.0 124

new objects

- Acrobat 8.1 125

- Acrobat 9.0 124

new properties

- Acrobat 8.0 126

- Acrobat 8.1 125

- Acrobat 9.0 124

nextSibling property 69

## Node object

- about 69
- detachFromCurrentAnimation method 70
- directionLocal property 48
- firstChild property 69
- hitEnabled property 69

info property 69

metadataString property 69

nextSibling property 69

opacity property 69

parent property 69

transform property 69

visible property 70

wireframeColor property 70

node property 106

nodes property 94

normalize method 73, 119

## O

objectGUID property 100

objectID property 100

objects

- deprecated in Acrobat 8.1 126

- new in Acrobat 8.1 125

- new in Acrobat 9.0 124

onEvent method 25, 34, 46, 63, 68, 76, 105, 107, 109, 113, 115

onMouseDown property 67

onMouseHit property 67

onMouseMove property 67

onMouseOut property 67

onMouseOver property 67

onMouseUp property 67

opacity property 36, 69

opacityTexture property 49

outerConeAngle property 48

outerRadius property 48

outlineAngle property 94

overrideFlyTool property 80

overrideNavTools property 80

overridePanTool property 81

overrideRotateTool property 81

overrideScrollWheel property 81

overrideSelection property 81

overrideSpinTool property 81

overrideViewChange property 81

overrideWalkTool property 81

overrideZoomTool property 81

## P

pan method 39

parent property 69

pause method 86

percentLoaded property 36

phongExponent property 49

play method 39, 86

playing property 36

pointsRenderModeColor property 77

position property 21, 42, 48

positionLocal property 21, 48

print method 31

println method 31

Procedural object 71

projectionType property 22, 24

properties

- corrections in Acrobat 9.0 125
- deprecated in Acrobat 8.1 126
- new in Acrobat 8.0 126
- new in Acrobat 8.1 125
- new in Acrobat 9.0 124
- undocumented in earlier versions 125

## Q

- quality property 36
- Quaternion method 72
- Quaternion object
  - about 72
  - interpolate method 73
  - interpolateInPlace method 73
  - normalize method 73
  - Quaternion method 72

## R

- readyState property 36
- reflectionStrength property 49
- reflectionTexture property 49
- refresh method 87
- remove method 27
- removeAll method 102
- removeByIndex method 102
- removeCustomMenuItem method 87
- removeCustomToolButton method 88
- removeEventHandler method 87
- removeItem method 103
- RENDER\_MODE\_BOUNDING\_BOX property 95
- RENDER\_MODE\_DEFAULT property 95
- RENDER\_MODE\_HIDDEN\_WIREFRAME property 96
- RENDER\_MODE\_ILLUSTRATION property 96
- RENDER\_MODE\_SHADED\_ILLUSTRATION property 96
- RENDER\_MODE\_SHADED\_VERTICES property 95
- RENDER\_MODE\_SHADED\_WIREFRAME property 95
- RENDER\_MODE\_SOLID property 95
- RENDER\_MODE\_SOLID\_WIREFRAME property 96
- RENDER\_MODE\_SOLID\_OUTLINE property 96
- RENDER\_MODE\_TRANSPARENT property 96
- RENDER\_MODE\_TRANSPARENT\_BOUNDING\_BOX property 95
- RENDER\_MODE\_TRANSPARENT\_BOUNDING\_BOX\_OUTLINE property 95
- RENDER\_MODE\_TRANSPARENT\_WIREFRAME property 96
- RENDER\_MODE\_VERTICES property 95
- RENDER\_MODE\_WIREFRAME property 95
- renderDoubleSided property 94
- RenderEvent object
  - about 75
  - canvas property 75
  - canvasPixelHeight property 75
  - canvasPixelWidth property 75
  - currentTool property 75
- RenderEventHandler method 76
- RenderEventHandler object
  - about 76
  - onEvent method 76
  - RenderEventHandler method 76
- renderMode property 64, 94

- RenderOptions object
  - about 77
  - boundingBoxColor property 77
  - clippingPlaneColor property 77
  - clippingPlaneIntersectionColor property 77
  - defaultAmbientColor property 77
  - defaultDiffuseColor property 77
  - defaultEmissiveColor property 77
  - defaultSpecularColor property 77
  - illustrationRenderModeFaceColor property 77
  - illustrationRenderModeLineColor property 77
  - pointsRenderModeColor property 77
  - shadedIllustrationRenderModeLineColor property 77
  - solidGridColorEven property 78
  - solidGridColorOdd property 78
  - solidRenderModeLineColor property 78
  - transparentBoundsRenderModeColor property 78
  - transparentGridColorEven property 78
  - transparentGridColorOdd property 78
  - wireframeRenderModeColor property 78
  - xAxisColor property 78
  - yAxisColor property 78
  - zAxisColor property 78
- reportAllTargets property 68
- RESOLUTION\_TYPE\_CUSTOM property 36
- RESOLUTION\_TYPE\_MOVIE property 36
- RESOLUTION\_TYPE\_WINDOW property 36
- resolutionType property 36
- Resource method 79
- Resource object
  - about 79
  - Resource method 79
  - type property 79
  - TYPE\_IMAGE property 79
  - TYPE\_MODEL property 79
  - TYPE\_UNKNOWN property 79
- rewind method 39
- rightButtonDown property 66
- roll property 22
- rotateAboutLine method 53
- rotateAboutLineInPlace method 54
- rotateAboutVector method 55
- rotateAboutVectorInPlace method 55
- rotateAboutX method 54
- rotateAboutXInPlace method 55
- rotateAboutY method 56
- rotateAboutYInPlace method 56
- rotateAboutZ method 56
- rotateAboutZInPlace method 57
- rotateWithQuaternion method 53
- rotateWithQuaternionInPlace method 53
- Runtime object
  - about 80
  - addCustomMenuItem method 83
  - addCustomToolButton method 83
  - addEventHandler method 84
  - BUTTON\_TYPE\_PUSH property 80
  - BUTTON\_TYPE\_TOOL property 80
  - canvasCount property 80
  - ctrlKeyDown property 80

- disableTool method 84
- enableTool method 84
- eventHandlerCount property 80
- getEventHandler method 85
- getRendererName method 85
- instances property 80
- MENU\_ITEM\_TYPE\_CHECKED property 80
- MENU\_ITEM\_TYPE\_DEFAULT property 80
- overrideFlyTools property 80
- overrideNavTools property 80
- overridePanTool property 81
- overrideRotateTool property 81
- overrideScrollWheel property 81
- overrideSelection property 81
- overrideSpinTool property 81
- overrideViewChange property 81
- overrideWalkTool property 81
- overrideZoomTool property 81
- pause method 86
- play method 86
- refresh method 87
- removeCustomMenuItem method 87
- removeCustomToolButton method 88
- removeEventHandler method 87
- scrollWheelSpeed property 81
- setCurrentTool method 88
- setCustomMenuItemChecked method 88
- shiftKeyDown property 81
- speedThreshold property 82
- strafeSpeed property 82
- TOOL\_NAME\_MEASURE property 82
- TOOL\_NAME\_PAN property 82
- TOOL\_NAME\_ROTATE property 82
- TOOL\_NAME\_SPIN property 82
- TOOL\_NAME\_WALK property 82
- TOOL\_NAME\_ZOOM property 82
- version property 82
- walkSpeed property 83

## S

- scale method 57, 120
- SCALE\_MODE\_EXACT\_FIT property 36
- SCALE\_MODE\_NO\_BORDER property 36
- SCALE\_MODE\_SHOW\_ALL property 37
- scaleComponent property 51
- scaleInPlace method 58, 120
- scaleMode property 36
- Scene object
  - about 91
  - activateAnimation method 97
  - addFlashForeground method 97
  - addModel method 98
  - ambientIlluminationColor property 91
  - animations property 91
  - cameras property 91
  - computeBoundingBox method 99
  - createClippingPlane method 98
  - createLight method 98
  - createSquareMesh method 98

- defaultRenderOptions property 91
- GRID\_MODE\_OFF property 91
- GRID\_MODE\_SOLID property 91
- GRID\_MODE\_TRANSPARENT property 91
- GRID\_MODE\_WIRE property 92
- gridMode property 91
- gridSize property 92
- lengthUnits property 92
- LIGHT\_MODE\_BLUE property 92
- LIGHT\_MODE\_BRIGHT property 92
- LIGHT\_MODE\_CAD property 93
- LIGHT\_MODE\_CUBE property 93
- LIGHT\_MODE\_DAY property 92
- LIGHT\_MODE\_FILE property 92
- LIGHT\_MODE\_HEADLAMP property 93
- LIGHT\_MODE\_NIGHT property 92
- LIGHT\_MODE\_NONE property 92
- LIGHT\_MODE\_RED property 92
- LIGHT\_MODE\_RGB property 92
- LIGHT\_MODE\_WHITE property 92
- lights property 93
- lightScaleFactor property 93
- lightScheme property 93
- materials property 93
- meshes property 93
- nodes property 94
- outlineAngle property 94
- RENDER\_MODE\_BOUNDING\_BOX property 95
- RENDER\_MODE\_DEFAULT property 95
- RENDER\_MODE\_HIDDEN\_WIREFRAME property 96
- RENDER\_MODE\_ILLUSTRATION property 96
- RENDER\_MODE\_SHADED\_ILLUSTRATION property 96
- RENDER\_MODE\_SHADED\_VERTICES property 95
- RENDER\_MODE\_SHADED\_WIREFRAME property 95
- RENDER\_MODE\_SOLID property 95
- RENDER\_MODE\_SOLID\_WIREFRAME property 96
- RENDER\_MODE\_SOLID\_OUTLINE property 96
- RENDER\_MODE\_TRANSPARENT property 96
- RENDER\_MODE\_TRANSPARENT\_BOUNDING\_BOX property 95
- RENDER\_MODE\_TRANSPARENT\_BOUNDING\_BOX\_OUTLINE property 95
- RENDER\_MODE\_TRANSPARENT\_WIREFRAME property 96
- RENDER\_MODE\_VERTICES property 95
- RENDER\_MODE\_WIREFRAME property 95
- renderDoubleSided property 94
- renderMode property 94
- selectedNode property 96
- showAxes property 96
- showGrid property 94
- smoothing property 97
- smoothingAngle property 97
- smoothingOverride property 97
- update method 99
- SceneObject object
  - about 100
  - name property 100
  - objectGUID property 100
  - objectId property 100
- SceneObjectList object 101
  - about 101

- count property 101
- getByGUID method 101
- getByID method 101
- getByIndex method 101
- getByName method 102
- removeAll method 102
- removeByIndex method 102
- removeItem method 103
- ScrollWheelEvent object
  - about 104
  - canvas property 104
  - canvasPxeleight property 104
  - canvasPixelWidth property 104
  - ctrlKeyDown property 104
  - currentTool property 104
  - deltaY property 104
  - shiftKeyDown property 104
- ScrollWheelEventHandler method 105
- ScrollWheelEventHandler object
  - about 105
  - onEvent method 105
  - ScrollWheelEventHandler method 105
- scrollWheelSpeed property 81
- selected property 106
- selectedNode property 96
- SelectionEvent object
  - about 106
  - node property 106
  - selected property 106
- SelectionEventHandler method 107
- SelectionEventHandler object
  - about 107
  - onEvent method 107
  - SelectionEventHandler method 107
- set method 28, 58, 59, 120, 121
- set3 method 29, 121
- setCamera method 26
- setColor method 17
- setCurrentTool method 88
- setCustomMenuItemChecked method 88
- setIdentity method 59
- setImage method 18, 111
- setVariable method 40
- setView method 59, 89
- setZoomRect method 40
- shadedIllustrationRenderModeLineColor property 77
- shiftKeyDown property 45, 66, 81, 104
- showAxes property 96
- showGrid property 94
- smoothing property 97
- smoothingAngle property 97
- smoothingOverride property 97
- solidGridColorEven property 78
- solidGridColorOdd property 78
- solidRenderModeLineColor property 78
- specularColor property 50
- specularStrength property 50
- speedThreshold property 82
- startTime property 16
- StateEvent object

- about 108
- stateString property 108
- type property 108
- TYPE\_LOAD property 108
- TYPE\_SAVE property 108
- StateEventHandler method 109
- StateEventHandler object
  - about 109
  - onEvent method 109
  - StateEventHandler method 109
- stateString property 108
- stop method 41
- strafeSpeed property 82
- subtract method 122
- subtractInPlace method 122
- surfaceNormal property 42

## T

- target property 22, 33, 34, 42, 68
- targetDistance property 24
- targetPosition property 22
- targetPositionLocal property 22
- Texture object
  - about 110
  - amount property 110
  - angle property 110
  - clampU property 110
  - clampV property 110
  - getImage method 110
  - image property 110
  - modulate property 110
  - setImage method 111
  - uOffset property 110
  - uScale property 110
  - use3DStyleMapping property 110
  - vOffset property 110
  - vScale property 110
- textureCoordinate property 42
- time event
  - measuring using the JavaScriptDate object 14
- time property 112
- TimeEvent object
  - about 112
  - deltaTime property 112
  - time property 112
- TimeEventHandler method 113
- TimeEventHandler object
  - about 113
  - onEvent method 113
  - TimeEventHandler method 113
- TOOL\_NAME\_FLY property 82
- TOOL\_NAME\_MEASURE property 82
- TOOL\_NAME\_PAN property 82
- TOOL\_NAME\_ROTATE property 82
- TOOL\_NAME\_SPIN property 82
- TOOL\_NAME\_WALK property 82
- TOOL\_NAME\_ZOOM property 82
- ToolEvent object
  - about 114



- canvas property 114
- canvasPixelHeight property 114
- canvasPixelWidth property 114
- currentTool property 114
- toolName property 114
- ToolEventHandler method 115
- ToolEventHandler object
  - about 115
  - onEvent method 115
  - ToolEventHandler method 115
- toolName property 114
- totalFrames property 37
- transform property 24, 69
- transformDirection method 60
- transformPosition method 60
- translate method 60
- translateInPlace method 61
- translation property 51
- transparentBoundsRenderModeColor property 78
- transparentGridColorEven property 78
- transparentGridColorOdd property 78
- transpose property 51
- transposeInPlace method 61
- type property 33, 48, 79, 108
- TYPE\_COMMAND property 33
- TYPE\_IMAGE property 79
- TYPE\_INFINITE property 48
- TYPE\_LOAD property 108
- TYPE\_MODEL property 79
- TYPE\_ORTHOGRAPHIC property 22
- TYPE\_PERSPECTIVE property 22
- TYPE\_POINT property 48
- TYPE\_PROGRESS property 33
- TYPE\_SAVE property 108
- TYPE\_SPOT property 48
- TYPE\_STATECHANGE property 33
- TYPE\_UNKNOWN property 79

## U

- uOffset property 110
- up property 22
- update method 99
- upLocal property 22
- uScale property 110
- use3DStyleMapping property 110

## V

- value property 33

- Vector3 method 116
- Vector3 object
  - about 116
  - add method 117
  - addInPlace method 117
  - addScaled method 117
  - addScaledInPlace method 118
  - blend method 118
  - blendInPlace method 118
  - cross method 119
  - dot method 119
  - length property 116
  - normalize method 119
  - scale method 120
  - scaleInPlace method 120
  - set method 120, 121
  - set3 method 121
  - subtract method 122
  - subtractInPlace method 122
- Vector3 method 116
- x property 116
- y property 116
- z property 116
- version property 82
- viewPlaneSize property 22, 24
- visible property 70
- vOffset property 110
- vScale property 110

## W

- walkSpeed property 83
- width property 43
- wireframeColor property 70
- wireframeRenderModeColor property 78

## X

- x property 37, 116
- xAxisColor property 78

## Y

- y property 37, 116
- yAxisColor property 78

## Z

- z property 116
- zAxisColor property 78
- zoom method 41