Contents

Preface ......................................................................................................................................................................... 8
  What’s in this guide? ...................................................................................................................................................... 8
  Who should read this guide? ......................................................................................................................................... 8
  Related documentation ................................................................................................................................................ 8

1 Introduction .............................................................................................................................................................. 9
  Object overview ......................................................................................................................................................... 10
  Basic objects ............................................................................................................................................................ 10
  Scene object ........................................................................................................................................................... 10
  Canvas object .......................................................................................................................................................... 10
  Runtime object .......................................................................................................................................................... 11
  Console object ......................................................................................................................................................... 11
  Resource objects .................................................................................................................................................... 11
  Event handlers ......................................................................................................................................................... 11
  CameraEvent .......................................................................................................................................................... 11
  KeyEvent ................................................................................................................................................................. 12
  MouseEvent ............................................................................................................................................................. 12
  RenderEvent ........................................................................................................................................................... 12
  ScrollWheelEvent .................................................................................................................................................... 13
  SelectionEvent ......................................................................................................................................................... 13
  TimeEvent ............................................................................................................................................................... 13
  ToolEvent ............................................................................................................................................................... 13

2 JavaScript Objects for Acrobat 3D .......................................................................................................................... 14
  Animation ................................................................................................................................................................. 15
  Background .............................................................................................................................................................. 16
  getColor ................................................................................................................................................................. 16
  getImage ............................................................................................................................................................... 16
  setColor ............................................................................................................................................................... 16
  setImage ............................................................................................................................................................... 17
  Bone ........................................................................................................................................................................ 18
  BoundingBox ........................................................................................................................................................... 19
  Camera .................................................................................................................................................................... 20
  getScreenFromPosition ........................................................................................................................................... 21
  getDirectionFromScreen ......................................................................................................................................... 22
  CameraEvent .......................................................................................................................................................... 23
  CameraEventHandler ................................................................................................................................................. 24
  CameraEventHandler ................................................................................................................................................. 24
  onEvent ................................................................................................................................................................. 24
  Canvas .................................................................................................................................................................... 25
  getCamera .............................................................................................................................................................. 25
  setCamera .............................................................................................................................................................. 25
  ClippingPlane ......................................................................................................................................................... 26
  remove ................................................................................................................................................................. 26
# 2 JavaScript Objects for Acrobat 3D (Continued)

<table>
<thead>
<tr>
<th>Class</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td>27</td>
</tr>
<tr>
<td>Color</td>
<td>27</td>
</tr>
<tr>
<td>set</td>
<td>27</td>
</tr>
<tr>
<td>set</td>
<td>28</td>
</tr>
<tr>
<td>set3</td>
<td>28</td>
</tr>
<tr>
<td>Console</td>
<td>29</td>
</tr>
<tr>
<td>print</td>
<td>29</td>
</tr>
<tr>
<td>println</td>
<td>29</td>
</tr>
<tr>
<td>Dummy</td>
<td>30</td>
</tr>
<tr>
<td>HitInfo</td>
<td>31</td>
</tr>
<tr>
<td>Host</td>
<td>32</td>
</tr>
<tr>
<td>Image</td>
<td>33</td>
</tr>
<tr>
<td>Image</td>
<td>33</td>
</tr>
<tr>
<td>KeyEvent</td>
<td>34</td>
</tr>
<tr>
<td>KeyEventHandler</td>
<td>36</td>
</tr>
<tr>
<td>KeyEvent</td>
<td>36</td>
</tr>
<tr>
<td>KeyEventHandler</td>
<td>36</td>
</tr>
<tr>
<td>onEvent</td>
<td>36</td>
</tr>
<tr>
<td>Light</td>
<td>37</td>
</tr>
<tr>
<td>Material</td>
<td>39</td>
</tr>
<tr>
<td>Matrix4x4</td>
<td>40</td>
</tr>
<tr>
<td>Matrix4x4</td>
<td>40</td>
</tr>
<tr>
<td>invertInPlace</td>
<td>40</td>
</tr>
<tr>
<td>isEqual</td>
<td>41</td>
</tr>
<tr>
<td>multiply</td>
<td>41</td>
</tr>
<tr>
<td>multiplyInPlace</td>
<td>41</td>
</tr>
<tr>
<td>rotateWithQuaternion</td>
<td>42</td>
</tr>
<tr>
<td>rotateWithQuaternionInPlace</td>
<td>42</td>
</tr>
<tr>
<td>rotateAboutLine</td>
<td>42</td>
</tr>
<tr>
<td>rotateAboutLineInPlace</td>
<td>43</td>
</tr>
<tr>
<td>rotateAboutX</td>
<td>43</td>
</tr>
<tr>
<td>rotateAboutXInPlace</td>
<td>44</td>
</tr>
<tr>
<td>rotateAboutVector</td>
<td>44</td>
</tr>
<tr>
<td>rotateAboutVectorInPlace</td>
<td>44</td>
</tr>
<tr>
<td>rotateAboutY</td>
<td>45</td>
</tr>
<tr>
<td>rotateAboutYInPlace</td>
<td>45</td>
</tr>
<tr>
<td>rotateAboutZ</td>
<td>45</td>
</tr>
<tr>
<td>rotateAboutZInPlace</td>
<td>46</td>
</tr>
<tr>
<td>scale</td>
<td>46</td>
</tr>
<tr>
<td>scaleInPlace</td>
<td>46</td>
</tr>
<tr>
<td>set</td>
<td>47</td>
</tr>
<tr>
<td>setIdentity</td>
<td>47</td>
</tr>
<tr>
<td>setView</td>
<td>47</td>
</tr>
<tr>
<td>transformDirection</td>
<td>48</td>
</tr>
<tr>
<td>transformPosition</td>
<td>48</td>
</tr>
<tr>
<td>translate</td>
<td>49</td>
</tr>
<tr>
<td>translateInPlace</td>
<td>49</td>
</tr>
<tr>
<td>transposeInPlace</td>
<td>49</td>
</tr>
</tbody>
</table>
## 2 JavaScript Objects for Acrobat 3D (Continued)

<table>
<thead>
<tr>
<th>Object/Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SceneObjectList</td>
<td>84</td>
</tr>
<tr>
<td>getByGUID</td>
<td>84</td>
</tr>
<tr>
<td>getByName</td>
<td>84</td>
</tr>
<tr>
<td>getByIndex</td>
<td>84</td>
</tr>
<tr>
<td>getByName</td>
<td>85</td>
</tr>
<tr>
<td>removeAll</td>
<td>86</td>
</tr>
<tr>
<td>removeByIndex</td>
<td>85</td>
</tr>
<tr>
<td>removeItem</td>
<td>85</td>
</tr>
<tr>
<td>ScrollWheelEvent</td>
<td>87</td>
</tr>
<tr>
<td>ScrollWheelEventHandler</td>
<td>88</td>
</tr>
<tr>
<td>TimeEventHandler</td>
<td>88</td>
</tr>
<tr>
<td>SelectionEvent</td>
<td>89</td>
</tr>
<tr>
<td>SelectionEventHandler</td>
<td>90</td>
</tr>
<tr>
<td>onEvent</td>
<td>90</td>
</tr>
<tr>
<td>Texture</td>
<td>91</td>
</tr>
<tr>
<td>getImage</td>
<td>91</td>
</tr>
<tr>
<td>setImage</td>
<td>91</td>
</tr>
<tr>
<td>TimeEvent</td>
<td>92</td>
</tr>
<tr>
<td>TimeEventHandler</td>
<td>92</td>
</tr>
<tr>
<td>onEvent</td>
<td>92</td>
</tr>
<tr>
<td>ToolEvent</td>
<td>93</td>
</tr>
<tr>
<td>ToolEventHandler</td>
<td>93</td>
</tr>
<tr>
<td>onEvent</td>
<td>93</td>
</tr>
<tr>
<td>Vector3</td>
<td>94</td>
</tr>
<tr>
<td>add</td>
<td>94</td>
</tr>
<tr>
<td>addInPlace</td>
<td>94</td>
</tr>
<tr>
<td>addScaled</td>
<td>94</td>
</tr>
<tr>
<td>addScaledInPlace</td>
<td>95</td>
</tr>
<tr>
<td>blend</td>
<td>95</td>
</tr>
<tr>
<td>blendInPlace</td>
<td>95</td>
</tr>
<tr>
<td>cross</td>
<td>96</td>
</tr>
<tr>
<td>normalize</td>
<td>96</td>
</tr>
<tr>
<td>dot</td>
<td>96</td>
</tr>
<tr>
<td>scale</td>
<td>96</td>
</tr>
<tr>
<td>scaleInPlace</td>
<td>96</td>
</tr>
<tr>
<td>set</td>
<td>97</td>
</tr>
<tr>
<td>set3</td>
<td>97</td>
</tr>
<tr>
<td>subtract</td>
<td>98</td>
</tr>
<tr>
<td>subtractInPlace</td>
<td>98</td>
</tr>
<tr>
<td>subtractInPlace</td>
<td>99</td>
</tr>
<tr>
<td>subtractInPlace</td>
<td>100</td>
</tr>
<tr>
<td>subtractInPlace</td>
<td>101</td>
</tr>
<tr>
<td>subtractInPlace</td>
<td>101</td>
</tr>
<tr>
<td>subtractInPlace</td>
<td>102</td>
</tr>
<tr>
<td>subtractInPlace</td>
<td>102</td>
</tr>
<tr>
<td>subtractInPlace</td>
<td>103</td>
</tr>
</tbody>
</table>
3 New Features and Changes.................................................................104
   Acrobat 8.1 changes........................................................................104
   New objects ......................................................................................104
   Additional properties in existing objects ......................................104
   Deprecated objects or properties ..................................................104
   Acrobat 8.0 changes........................................................................104
   Additional properties in existing objects ......................................105

Index .....................................................................................................106
Preface

The JavaScript™ API allows you to 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 Family Developer Center, http://www.adobe.com/go/acrobat_developer.

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

To create 3D annotations and to attach scripts to them using this API, you will need Adobe® Acrobat®
Professional and Acrobat 3D. Scripts attached to 3D annotations can run on Acrobat Professional,
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
details 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".

```javascript
// 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.

```javascript
// Get the Annot3D script context of the targeted annot.
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.

```javascript
function setRenderMode( renderModeName ) {
    for ( var i=0; i < scene.meshes.count; i++ ) {
        scene.meshes.getViewById(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 on page 75.

Canvas object

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

For more information, see Canvas on page 25.
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" on page 67.

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:

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

The `Resource` and `Image` objects are covered on page 66 and page 33, respectively.

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, page 71, of the `Runtime` object.

CamaraEvent

A CamaraEvent is created when a View is selected.

For information, see `CameraEvent on page 23`. 
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:

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

For information, see [KeyEvent on page 34](#).

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 on page 50](#).

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:

```javascript
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 on page 53](#).

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 on page 62](#).
ScrollWheelEvent

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

For more information, see ScrollWheelEvent on page 87.

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 on page 89.

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:

```javascript
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 on the TimeEvent, see page 93.

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” on page 95.
This chapter describes the following 3D JavaScript objects:

<table>
<thead>
<tr>
<th>Animation</th>
<th>MouseEvent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
<td>MouseEventHandler</td>
</tr>
<tr>
<td>Bone</td>
<td>Node</td>
</tr>
<tr>
<td>BoundingBox</td>
<td>Procedural</td>
</tr>
<tr>
<td>Camera</td>
<td>Quaternion</td>
</tr>
<tr>
<td>CameraEvent</td>
<td>RenderEvent</td>
</tr>
<tr>
<td>CameraEventHandler</td>
<td>RenderEventHandler</td>
</tr>
<tr>
<td>Canvas</td>
<td>RenderOptions</td>
</tr>
<tr>
<td>ClippingPlane</td>
<td>Resource</td>
</tr>
<tr>
<td>Color</td>
<td>Runtime</td>
</tr>
<tr>
<td>Console</td>
<td>Scene</td>
</tr>
<tr>
<td>Dummy</td>
<td>SceneObject</td>
</tr>
<tr>
<td>HitInfo</td>
<td>SceneObjectList</td>
</tr>
<tr>
<td>Host</td>
<td>ScrollWheelEvent</td>
</tr>
<tr>
<td>Image</td>
<td>ScrollWheelEventHandler</td>
</tr>
<tr>
<td>KeyEvent</td>
<td>SelectionEvent</td>
</tr>
<tr>
<td>KeyEventHandler</td>
<td>SelectionEventHandler</td>
</tr>
<tr>
<td>Light</td>
<td>Texture</td>
</tr>
<tr>
<td>Material</td>
<td>TimeEvent</td>
</tr>
<tr>
<td>Matrix4x4</td>
<td>TimeEventHandler</td>
</tr>
<tr>
<td>MenuEvent</td>
<td>ToolEvent</td>
</tr>
<tr>
<td>MenuEventHandler</td>
<td>ToolEventHandler</td>
</tr>
<tr>
<td>Mesh</td>
<td>Vector3</td>
</tr>
</tbody>
</table>

**Note:** A property labeled as read-only is one whose value cannot be set. An object labeled as 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 are assumed to have read/write access.
Animation

A type of `SceneObject`, page 83, 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

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>currentTime</td>
<td>number</td>
<td></td>
<td>The current time measured in seconds.</td>
</tr>
<tr>
<td>endTime</td>
<td>number</td>
<td>read-only</td>
<td>The end time of the sequence, measured in seconds.</td>
</tr>
<tr>
<td>framesPerSecond</td>
<td>number</td>
<td>read-only</td>
<td>The number of frames per second used to author the sequence.</td>
</tr>
<tr>
<td>length</td>
<td>number</td>
<td>read-only</td>
<td>The length of the <code>Animation</code>, measured in seconds.</td>
</tr>
<tr>
<td>startTime</td>
<td>number</td>
<td>read-only</td>
<td>The start time of the sequence, measured in seconds.</td>
</tr>
</tbody>
</table>
Background

Represents the background of a Canvas. It can be used as a target of a MouseEventHandler.

For information on the Canvas and MouseEventHandler, see page 25 and page 55, respectively.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>image</td>
<td>Image</td>
<td></td>
<td>Acrobat 7.0.7 The Image to be used by the Background.</td>
</tr>
</tbody>
</table>

getColor

Obtains the background Color.

Syntax

g getColor()

Returns

A Color object representing the background color of the Canvas.

gImage

Deprecated

Obtains the background Image.

Syntax

g 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 will be 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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>topColor</td>
<td>A Color object representing the desired background color. If bottomColor is used, topColor represents the top background color used in a linear gradient.</td>
</tr>
<tr>
<td>bottomColor</td>
<td>(Optional) A Color object representing the bottom background color used in a linear gradient.</td>
</tr>
</tbody>
</table>

Returns

undefined

**setImage**

Deprecated

Sets the background Image.

**Syntax**

```
setImage(image)
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>image</td>
<td>An Image object representing the desired background image.</td>
</tr>
</tbody>
</table>

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 on page 57 and Mesh on page 52.
BoundingBox

Represents an axis-aligned bounding box.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
</table>
| center   | Vector3 | read-only | Acrobat 7.0.7  
The coordinates of the BoundingBox center.                                 |
| max      | Vector3 | read-only | The coordinates of the BoundingBox corner with the greatest x, y, and z values. |
| min      | Vector3 | read-only | 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. (See Node on page 57.)

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>binding</td>
<td>string</td>
<td></td>
<td>The view plane calculation type, which can take one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;min&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;max&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;horizontal&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;vertical&quot;</td>
</tr>
<tr>
<td>BINDING_HORIZONTAL</td>
<td>string</td>
<td></td>
<td>Acrobat 7.0.7. A string constant for the binding value of &quot;horizontal&quot;.</td>
</tr>
<tr>
<td>BINDING_MAX</td>
<td>string</td>
<td></td>
<td>Acrobat 7.0.7. A string constant for the binding value of &quot;max&quot;.</td>
</tr>
<tr>
<td>BINDING_MIN</td>
<td>string</td>
<td></td>
<td>Acrobat 7.0.7. A string constant for the binding value of &quot;min&quot;.</td>
</tr>
<tr>
<td>BINDING_VERTICAL</td>
<td>string</td>
<td></td>
<td>Acrobat 7.0.7. A string constant for the binding value of &quot;vertical&quot;.</td>
</tr>
<tr>
<td>far</td>
<td>number</td>
<td></td>
<td>The distance from the Camera to the far clipping plane. A value of -1 for both near and far signifies to use auto clipping plane calculations.</td>
</tr>
<tr>
<td>fov</td>
<td>number</td>
<td></td>
<td>The size of the field of view for perspective Camera objects, measured in radians.</td>
</tr>
<tr>
<td>near</td>
<td>number</td>
<td></td>
<td>The distance from the Camera to the near clipping plane. A value of -1 for both near and far signifies to use auto clipping plane calculations.</td>
</tr>
<tr>
<td>position</td>
<td>Vector3</td>
<td></td>
<td>The position of the origin of the Camera in world space.</td>
</tr>
<tr>
<td>positionLocal</td>
<td>Vector3</td>
<td></td>
<td>The position of the origin of the Camera in local space.</td>
</tr>
<tr>
<td>projectionType</td>
<td>string</td>
<td></td>
<td>The type of projection, which can take one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;perspective&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;orthographic&quot;</td>
</tr>
</tbody>
</table>
getScreenFromPosition

Obtains the screen coordinates of the provided 3D position.

Syntax

getScreenFromPosition(position, canvasWidth, canvasHeight)

Parameters

<table>
<thead>
<tr>
<th>Position</th>
<th>Description</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>position</td>
<td>A Vector3 object representing the 3D position.</td>
<td>read-only</td>
</tr>
<tr>
<td>canvasWidth</td>
<td>The width of the Canvas, measured in pixels.</td>
<td>read-only</td>
</tr>
<tr>
<td>canvasHeight</td>
<td>The height of the Canvas, measured in pixels.</td>
<td>read-only</td>
</tr>
</tbody>
</table>

Returns

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

See “Vector3” on page 97 for more information on the return object.
getDirectionFromScreen

Obtains the direction from the normalized coordinates

Syntax

getDirectionFromScreen(x, y, canvasWidth, canvasHeight)

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>The x-coordinate, measured in pixels.</td>
</tr>
<tr>
<td>y</td>
<td>The y-coordinate, measured in pixels.</td>
</tr>
<tr>
<td>canvasWidth</td>
<td>The width of the Canvas, measured in pixels.</td>
</tr>
<tr>
<td>canvasHeight</td>
<td>The height of the Canvas, measured in pixels.</td>
</tr>
</tbody>
</table>

Returns

A Vector3 object representing the direction

See “Vector3” on page 97 for more information on the return object.
CameraEvent

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

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>binding</td>
<td>string</td>
<td>read-only</td>
<td>The view plane calculation type, which can take one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;min&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;max&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;horizontal&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;vertical&quot;</td>
</tr>
<tr>
<td>canvas</td>
<td>Canvas</td>
<td>read-only</td>
<td>The <code>Canvas</code> in which the event took place.</td>
</tr>
<tr>
<td>currentTool</td>
<td>string</td>
<td>read-only</td>
<td>The name of the current tool.</td>
</tr>
<tr>
<td>far</td>
<td>number</td>
<td>read-only</td>
<td>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.</td>
</tr>
<tr>
<td>fov</td>
<td>number</td>
<td>read-only</td>
<td>The size of the field of view for perspective <code>Camera</code> objects, measured in radians.</td>
</tr>
<tr>
<td>isNewCanvas</td>
<td>Boolean</td>
<td>read-only</td>
<td>Deprecated</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Determines whether this is the first event for this <code>Canvas</code>.</td>
</tr>
<tr>
<td>near</td>
<td>number</td>
<td>read-only</td>
<td>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.</td>
</tr>
<tr>
<td>projectionType</td>
<td>string</td>
<td>read-only</td>
<td>The type of projection, which can take one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;perspective&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;orthographic&quot;</td>
</tr>
<tr>
<td>targetDistance</td>
<td>Vector3</td>
<td>read-only</td>
<td>The distance from the <code>Camera</code> to its target.</td>
</tr>
<tr>
<td>transform</td>
<td>Matrix4x4</td>
<td>read-only</td>
<td>The <code>Camera</code> object’s transformation matrix.</td>
</tr>
<tr>
<td>viewPlaneSize</td>
<td>number</td>
<td>read-only</td>
<td>The size of the view plane, measured in scene units.</td>
</tr>
</tbody>
</table>
CameraEventHandler

Exposes a callback mechanism that allows a function to be evaluated when an camera event occurs. Event handlers are registered with the Runtime `addEventHandler` method, page 71.

**CameraEventHandler**

**Constructor**

**Syntax**

```javascript
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**

```javascript
onEvent(event)
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>A <code>CameraEvent</code> object representing the event.</td>
</tr>
</tbody>
</table>

**Returns**

`undefined`
Canvas

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

See related objects, Scene on page 75 and Camera on page 20.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>background</td>
<td>Background</td>
<td>read-only</td>
<td>The Background object associated with the Canvas.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>camera</td>
<td>The Camera object used to set the object's value.</td>
</tr>
</tbody>
</table>

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, described on page 81.

**remove**

Removes the ClippingPlane object from the Scene.

**Syntax**

```javascript
remove()
```

**Returns**

undefined
Color

An object that represents a RGB encoded color.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>number</td>
<td>The blue component, which may contain a value from 0.0 to 1.0.</td>
</tr>
<tr>
<td>g</td>
<td>number</td>
<td>The green component, which may contain a value from 0.0 to 1.0.</td>
</tr>
<tr>
<td>r</td>
<td>number</td>
<td>The red component, which may contain a value from 0.0 to 1.0.</td>
</tr>
</tbody>
</table>

Color

Constructor

Syntax

new Color()

Returns

A Color object, initialized to black

Color

Constructor

Syntax

new Color(r, g, b)

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>The red component, which may contain a value from 0.0 to 1.0.</td>
</tr>
<tr>
<td>g</td>
<td>The green component, which may contain a value from 0.0 to 1.0.</td>
</tr>
<tr>
<td>b</td>
<td>The blue component, which may contain a value from 0.0 to 1.0.</td>
</tr>
</tbody>
</table>

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 Color object used to set the object’s value. |

Returns

undefined

**set**

Adobe Acrobat SDK

**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 may contain a value from 0.0 to 1.0. |
| g | The green component, which may contain a value from 0.0 to 1.0. |
| b | The blue component, which may contain 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 may contain a value from 0.0 to 1.0. |
| g | The green component, which may contain a value from 0.0 to 1.0. |
| b | The blue component, which may contain a value from 0.0 to 1.0. |

Returns

undefined
Console

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

**print**

Prints a string to the console.

**Syntax**

```
print(string)
```

**Parameters**

- `string` The text to be printed to the console.

**Returns**

undefined

**println**

Prints a string with an accompanying newline to the console.

**Syntax**

```
println(string)
```

**Parameters**

- `string` The text to be printed to the console.

**Returns**

undefined
**Dummy**

**Deprecated**

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

The object returned when a hit test occurs during a MouseEvent, page 50.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>distance</td>
<td>number</td>
<td>read-only</td>
<td>The distance from the Camera to the HitInfo object’s position.</td>
</tr>
</tbody>
</table>
| material     | Material   | read-only | Acrobat 8.1
The material of the node that was hit. |
| position     | Vector3    | read-only | The position of the point where the hit occurred.                          |
| surfaceNormal | Vector3    | read-only | Acrobat 8.1
World-space surface normal direction at hit location. |
| target       | Node       | read-only | The target of the hit test.                                                |
| textureCoordinate | Vector3    | read-only | 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 `Document` object in which the 3D annotation is contained.
Image

An object that represents an image.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>number</td>
<td>read-only</td>
<td>The image's height, measured in pixels.</td>
</tr>
<tr>
<td>width</td>
<td>number</td>
<td>read-only</td>
<td>The image's width, measured in pixels.</td>
</tr>
</tbody>
</table>

Image

Constructor

Syntax

```javascript
new Image(resource)
```

Parameters

<table>
<thead>
<tr>
<th>resource</th>
<th>An Image object used to create the new object.</th>
</tr>
</thead>
</table>

Returns

An Image object

See “Image” on page 33 for more information on the return object.
**KeyEvent**

An object that is passed as an argument to the `onEvent` method, page 24, of the `KeyEventHandler` object.

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>canvas</td>
<td>Canvas</td>
<td>read-only</td>
<td>The Canvas in which the <code>KeyEvent</code> took place</td>
</tr>
<tr>
<td>canvasPixelHeight</td>
<td>integer</td>
<td>read-only</td>
<td>The height, measured in pixels, of the Canvas</td>
</tr>
<tr>
<td>canvasPixelWidth</td>
<td>integer</td>
<td>read-only</td>
<td>The width, measured in pixels, of the Canvas</td>
</tr>
<tr>
<td>characterCode</td>
<td>integer</td>
<td>read-only</td>
<td>The value of the character pressed according to Acrobat’s character mapping, as per this listing of Acrobat character codes:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#</th>
<th>Keys</th>
<th>#</th>
<th>Keys</th>
<th>#</th>
<th>Keys</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>Left</td>
<td>65</td>
<td>A</td>
<td>97</td>
<td>a</td>
</tr>
<tr>
<td>29</td>
<td>Right</td>
<td>66</td>
<td>B</td>
<td>98</td>
<td>b</td>
</tr>
<tr>
<td>30</td>
<td>Down</td>
<td>67</td>
<td>C</td>
<td>99</td>
<td>c</td>
</tr>
<tr>
<td>31</td>
<td>Up</td>
<td>68</td>
<td>D</td>
<td>100</td>
<td>d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>69</td>
<td>E</td>
<td>101</td>
<td>e</td>
</tr>
<tr>
<td></td>
<td></td>
<td>70</td>
<td>F</td>
<td>102</td>
<td>f</td>
</tr>
<tr>
<td></td>
<td></td>
<td>71</td>
<td>G</td>
<td>103</td>
<td>g</td>
</tr>
<tr>
<td></td>
<td></td>
<td>72</td>
<td>H</td>
<td>104</td>
<td>h</td>
</tr>
<tr>
<td></td>
<td></td>
<td>73</td>
<td>I</td>
<td>105</td>
<td>i</td>
</tr>
<tr>
<td></td>
<td></td>
<td>74</td>
<td>J</td>
<td>106</td>
<td>j</td>
</tr>
<tr>
<td></td>
<td>Space</td>
<td>75</td>
<td>K</td>
<td>107</td>
<td>k</td>
</tr>
<tr>
<td></td>
<td></td>
<td>76</td>
<td>L</td>
<td>108</td>
<td>l</td>
</tr>
<tr>
<td></td>
<td></td>
<td>77</td>
<td>M</td>
<td>109</td>
<td>m</td>
</tr>
<tr>
<td></td>
<td></td>
<td>78</td>
<td>N</td>
<td>110</td>
<td>n</td>
</tr>
<tr>
<td></td>
<td></td>
<td>79</td>
<td>O</td>
<td>111</td>
<td>o</td>
</tr>
<tr>
<td></td>
<td></td>
<td>80</td>
<td>P</td>
<td>112</td>
<td>p</td>
</tr>
<tr>
<td></td>
<td></td>
<td>81</td>
<td>Q</td>
<td>113</td>
<td>q</td>
</tr>
<tr>
<td></td>
<td></td>
<td>82</td>
<td>R</td>
<td>114</td>
<td>r</td>
</tr>
<tr>
<td></td>
<td></td>
<td>83</td>
<td>S</td>
<td>115</td>
<td>s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>T</td>
<td>116</td>
<td>t</td>
</tr>
<tr>
<td></td>
<td></td>
<td>85</td>
<td>U</td>
<td>117</td>
<td>u</td>
</tr>
<tr>
<td></td>
<td></td>
<td>86</td>
<td>V</td>
<td>118</td>
<td>v</td>
</tr>
<tr>
<td></td>
<td></td>
<td>87</td>
<td>W</td>
<td>119</td>
<td>w</td>
</tr>
<tr>
<td></td>
<td></td>
<td>88</td>
<td>X</td>
<td>120</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td></td>
<td>89</td>
<td>Y</td>
<td>121</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td></td>
<td>90</td>
<td>Z</td>
<td>122</td>
<td>z</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Access</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>----------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctrlKeyDown</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the Ctrl key (Windows) or Command key (Mac OS) was pressed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> Acrobat will intercept many of the Ctrl + key events because they are used for accelerators in the main application.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>currentTool</td>
<td>string</td>
<td>read-only</td>
<td>The name of the current tool.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>shiftKeyDown</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the Shift key was pressed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> Holding the shift key down changes the value of the <code>KeyEvent.characterCode</code> property.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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 `addEventHandler` method, described on page 71.

**KeyEventHandler**

**Constructor**

**Syntax**

```javascript
new KeyEventHandler()
```

**Returns**

A `KeyEventHandler` object

**onEvent**

A method that is called when a key is pressed.

**Syntax**

```javascript
onEvent(event)
```

**Parameters**

| event | A `KeyEvent` 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 below, it also contains the same methods and properties as a Node.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attenuationA</td>
<td>number</td>
<td></td>
<td>The a coefficient for attenuationType &quot;abc&quot;.</td>
</tr>
<tr>
<td>attenuationB</td>
<td>number</td>
<td></td>
<td>The b coefficient for attenuationType &quot;abc&quot;.</td>
</tr>
<tr>
<td>attenuationC</td>
<td>number</td>
<td></td>
<td>The c coefficient for attenuationType &quot;abc&quot;.</td>
</tr>
</tbody>
</table>
| attenuationType| string  |        | The style of attenuation for the Light 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 d is the distance from the light. One of the following values may be assigned:  
  ● "abc"
  ● "none" |
| ATTENUATION_ABC| string  | read-only | Acrobat 7.0.7  
  A string constant for the attenuationType of "abc".                          |
| ATTENUATION_NONE| string  | read-only | Acrobat 7.0.7  
  A string constant for the attenuationType of "none".                           |
| brightness     | number  |        | Specifies the brightness of the emission from the Light. A value of 1 represents a brightness of 100%, though the property may be assigned higher values. |
| color          | Color   | read-only | Specifies the color of the light.                                          |
| direction      | Vector3 | read-only | The direction toward which the light is pointing.                           |
| directionLocal | Vector3 | read-only | Acrobat 7, but not documented until Acrobat 8.1  
  The direction toward which the light is pointing relative to its parent Node. |
<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>innerConeAngle</td>
<td>number</td>
<td></td>
<td>The angle, measured in radians, about the direction in which the light is of uniform full density.</td>
</tr>
<tr>
<td>innerRadius</td>
<td>number</td>
<td></td>
<td>The distance within which the light is of uniform full density.</td>
</tr>
<tr>
<td>outerConeAngle</td>
<td>number</td>
<td></td>
<td>The angle, measured in radians, about the direction outside of which the light's intensity is zero.</td>
</tr>
<tr>
<td>outerRadius</td>
<td>number</td>
<td></td>
<td>The distance beyond which the light's intensity is zero.</td>
</tr>
<tr>
<td>position</td>
<td>Vector3</td>
<td>read-only</td>
<td>The position of the Light object.</td>
</tr>
<tr>
<td>positionLocal</td>
<td>Vector3</td>
<td>read-only</td>
<td>The position of the Light object relative to its parent Node.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td></td>
<td>The type of Light object being represented. One of the following values may be assigned:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;point&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;spot&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;infinite&quot;</td>
</tr>
<tr>
<td>TYPE_INFINITE</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the Light type of &quot;infinite&quot;.</td>
</tr>
<tr>
<td>TYPE_POINT</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the Light type of &quot;point&quot;.</td>
</tr>
<tr>
<td>TYPE_SPOT</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the Light type of &quot;spot&quot;.</td>
</tr>
</tbody>
</table>
Material

A `SceneObject` that controls the appearance of materials using the fixed function shader. In addition to the properties below, it also contains the same methods and properties as a `SceneObject`, documented on page 83.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ambientColor</td>
<td>Color</td>
<td>read-only</td>
<td>The ambient color.</td>
</tr>
<tr>
<td>ambientTexture</td>
<td>Texture</td>
<td>read-only</td>
<td>The ambient texture.</td>
</tr>
<tr>
<td>bumpTexture</td>
<td>Texture</td>
<td>read-only</td>
<td>A texture map whose value is used to describe the roughness of the object.</td>
</tr>
<tr>
<td>diffuseColor</td>
<td>Color</td>
<td>read-only</td>
<td>The matte color of an object.</td>
</tr>
<tr>
<td>diffuseTexture</td>
<td>Texture</td>
<td>read-only</td>
<td>A texture map that is used for the matte color of the object.</td>
</tr>
<tr>
<td>emissiveColor</td>
<td>Color</td>
<td>read-only</td>
<td>The emissive color.</td>
</tr>
<tr>
<td>emissiveTexture</td>
<td>Texture</td>
<td>read-only</td>
<td>The emissive texture.</td>
</tr>
<tr>
<td>opacity</td>
<td>number</td>
<td></td>
<td>The total opacity of the material.</td>
</tr>
<tr>
<td>opacityTexture</td>
<td>Texture</td>
<td>read-only</td>
<td>A texture map whose brightness is used for the level of opacity of the object. White signifies completely opaque while black signifies completely transparent.</td>
</tr>
<tr>
<td>phongExponent</td>
<td>number</td>
<td></td>
<td>The phong exponent.</td>
</tr>
<tr>
<td>reflectionStrength</td>
<td>Number</td>
<td></td>
<td>The reflection level, which may contain a value from 0.0 to 1.0.</td>
</tr>
<tr>
<td>reflectionTexture</td>
<td>Texture</td>
<td>read-only</td>
<td>The reflection texture.</td>
</tr>
<tr>
<td>specularColor</td>
<td>Color</td>
<td>read-only</td>
<td>The specular color.</td>
</tr>
<tr>
<td>specularStrength</td>
<td>number</td>
<td></td>
<td>The specular strength, which is a measure of how shiny the material is.</td>
</tr>
</tbody>
</table>
Matrix4x4

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

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>determinant</td>
<td>number</td>
<td></td>
<td>The determinant of the matrix.</td>
</tr>
<tr>
<td>inverse</td>
<td>Matrix4x4</td>
<td>read-only</td>
<td>The inverse of the matrix.</td>
</tr>
<tr>
<td>scaleComponent</td>
<td>Vector3</td>
<td>read-only</td>
<td>The scale component of the transformation.</td>
</tr>
<tr>
<td>translation</td>
<td>Vector3</td>
<td>read-only</td>
<td>The translation component of the transformation.</td>
</tr>
<tr>
<td>transpose</td>
<td>Matrix4x4</td>
<td>read-only</td>
<td>The transpose of the matrix.</td>
</tr>
</tbody>
</table>

Matrix4x4

Constructor

```javascript
new Matrix4x4()
```

Returns

A `Matrix4x4` object initialized to the identity matrix

Matrix4x4

Constructor

```javascript
new Matrix4x4(matrix)
```

Parameters

| matrix       | A `Matrix4x4` 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

matrix A Matrix4x4 object used for the comparison.

Returns

True if the matrices are equal, false otherwise.

multiply

Multiplies the current matrix by the specified matrix.

Syntax

multiply(matrix)

Parameters

matrix A Matrix4x4 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

matrix A Matrix4x4 object used for the multiplication.
Returns

undefined

rotateWithQuaternion

Rotates the current matrix using the specified Quaternion

Syntax

rotateWithQuaternion(quaternion)

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quaternion</td>
<td>A Quaternion object used for the rotation.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quaternion</td>
<td>A Quaternion object used for the rotation.</td>
</tr>
</tbody>
</table>

Returns

undefined

rotateAboutLine

Rotates the current matrix about the specified line.

Syntax

rotateAboutLine(angle, start, end)
Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>angle</td>
<td>The angle of rotation, in radians</td>
</tr>
<tr>
<td>start</td>
<td>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>).</td>
</tr>
<tr>
<td>end</td>
<td>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>).</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>angle</td>
<td>The angle of rotation, in radians</td>
</tr>
<tr>
<td>start</td>
<td>A <code>Vector3</code> object used to specify the line of rotation (which is represented by <code>start - end</code>).</td>
</tr>
<tr>
<td>end</td>
<td>A <code>Vector3</code> object used to specify the line of rotation (which is represented by <code>start - end</code>).</td>
</tr>
</tbody>
</table>

Returns

`undefined`

`rotateAboutX`  
Rotates the current matrix about the x-axis.

Syntax

`rotateAboutX(angle)`

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>angle</td>
<td>The angle of rotation, in radians.</td>
</tr>
</tbody>
</table>
rotateAboutXInPlace

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

**Syntax**

```javascript
rotateAboutXInPlace(angle)
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>angle</td>
<td>The angle of rotation, in radians.</td>
</tr>
</tbody>
</table>

**Returns**

undefined

rotateAboutVector

Rotates the current matrix about the specified vector.

**Syntax**

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

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>angle</td>
<td>The angle of rotation, in radians.</td>
</tr>
<tr>
<td>axis</td>
<td>A Vector3 object about which the matrix is rotated.</td>
</tr>
</tbody>
</table>

**Returns**

A Matrix4x4 object

rotateAboutVectorInPlace

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

**Syntax**

```javascript
rotateAboutVectorInPlace(angle, axis)
```
Parameters

| angle   | The angle of rotation, in radians.
|---------|------------------------------------------------------------------
| axis    | A Vector3 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)
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**

A Matrix4x4 object

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

<table>
<thead>
<tr>
<th>x</th>
<th>The scaling component in the x-direction.</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>The scaling component in the y-direction.</td>
</tr>
<tr>
<td>z</td>
<td>The scaling component in the z-direction.</td>
</tr>
</tbody>
</table>

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

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

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

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>position</td>
<td>A Vector3 object used to specify the position component.</td>
</tr>
<tr>
<td>direction</td>
<td>A Vector3 object used to specify the direction component.</td>
</tr>
<tr>
<td>up</td>
<td>A Vector3 object used to specify the upward component.</td>
</tr>
</tbody>
</table>

**Returns**

undefined

**transformDirection**

Transforms the specified vector by the current matrix.

**Syntax**

```javascript
transformDirection(vector)
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vector</td>
<td>The Vector3 object to be transformed.</td>
</tr>
</tbody>
</table>

**Returns**

A Vector3 object

**transformPosition**

Transforms the specified position by the current matrix.

**Syntax**

```javascript
transformPosition(position)
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>position</td>
<td>A Vector3 object representing the position to be transformed.</td>
</tr>
</tbody>
</table>

**Returns**

A Vector3 object
translate

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

Syntax

translate(translation)

Parameters

| translation | The Vector3 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 Vector3 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()
MenuEvent

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

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>canvas</td>
<td>Canvas</td>
<td>read-only</td>
<td>The Canvas in which the MenuEvent took place.</td>
</tr>
<tr>
<td>currentTool</td>
<td>string</td>
<td>read-only</td>
<td>The name of the current tool.</td>
</tr>
<tr>
<td>menuItemChecked</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the menu item was selected.</td>
</tr>
<tr>
<td>menuItemName</td>
<td>string</td>
<td>read-only</td>
<td>The name of the selected menu item.</td>
</tr>
</tbody>
</table>
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

Constructor

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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>A MenuEvent object representing the event.</td>
</tr>
</tbody>
</table>

Returns

undefined
Mesh

A `Node` object that contains geometry. A `Mesh` object with no geometry will have children `Node` objects that may be transformed as a group. In addition to the methods and properties below, it also contains the same methods and properties as a `Node`, see page 57.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>material</td>
<td>Material</td>
<td>The <code>Mesh</code> object’s default <code>Material</code>.</td>
</tr>
<tr>
<td>renderMode</td>
<td>string</td>
<td>The <code>Mesh</code> object’s rendering style, which can be one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;default&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;bounding box&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;transparent bounding box&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;transparent bounding box outline&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;vertices&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;shaded vertices&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;wireframe&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;shaded wireframe&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;solid&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;transparent&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;solid wireframe&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;transparent wireframe&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;illustration&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;solid outline&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;shaded illustration&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● &quot;hidden wireframe&quot;</td>
</tr>
</tbody>
</table>
**MouseEvent**

An object that is passed as an argument to the `onEvent` method of the `MouseEventHandler` object, page 55.

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>canvas</td>
<td>Canvas</td>
<td>read-only</td>
<td>The Canvas in which the MouseEvent took place.</td>
</tr>
<tr>
<td>canvasPixelHeight</td>
<td>integer</td>
<td>read-only</td>
<td>The height, measured in pixels, of the Canvas in which the MouseEvent took place.</td>
</tr>
<tr>
<td>canvasPixelWidth</td>
<td>integer</td>
<td>read-only</td>
<td>The width, measured in pixels, of the Canvas in which the MouseEvent took place.</td>
</tr>
<tr>
<td>ctrlKeyDown</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the Ctrl key (Windows) or Command key (Mac OS) was pressed.</td>
</tr>
<tr>
<td>currentTool</td>
<td>string</td>
<td>read-only</td>
<td>The name of the current tool.</td>
</tr>
<tr>
<td>hits</td>
<td>Array</td>
<td>read-only</td>
<td>A set of HitInfo objects ordered by distance from nearest to furthest.</td>
</tr>
<tr>
<td>isDoubleClick</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the a double-click event occurred</td>
</tr>
<tr>
<td>isMouseDown</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the mouse button has been pressed</td>
</tr>
<tr>
<td>isMouseHit</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the target is under the mouse cursor.</td>
</tr>
<tr>
<td>isMouseMove</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the mouse position has changed.</td>
</tr>
<tr>
<td>isMouseOut</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the mouse position has moved off the target.</td>
</tr>
<tr>
<td>isMouseOver</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the mouse position has moved onto the target.</td>
</tr>
<tr>
<td>isMouseUp</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the mouse button has been released.</td>
</tr>
<tr>
<td>leftButtonDown</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the left mouse button has been pressed.</td>
</tr>
<tr>
<td>mouseX</td>
<td>integer</td>
<td>read-only</td>
<td>The x position of the mouse cursor in the Canvas.</td>
</tr>
<tr>
<td>mouseY</td>
<td>integer</td>
<td>read-only</td>
<td>The y position of the mouse cursor in the Canvas.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Access</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>---------</td>
<td>--------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>rightButtonDown</td>
<td>Boolean</td>
<td>read-only</td>
<td>Version 7.0.1&lt;br&gt;Determines whether the right mouse button has been pressed.</td>
</tr>
<tr>
<td>shiftKeyDown</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the Shift key has been pressed.</td>
</tr>
</tbody>
</table>
MouseEventHandler

An object that exposes a callback mechanism that allows a function to be evaluated when mouse event occurs. The handler may be customized to filter out certain event types. Event handlers are registered with the Runtime `addEventHandler` method, see page 71.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onMouseDoubleClick</td>
<td>Boolean</td>
<td></td>
<td>When set to true the handler will be called back when a mouse button is clicked twice on the target object in rapid succession. If no target is specified the handler will call back on any double-click.</td>
</tr>
<tr>
<td>onMouseDown</td>
<td>Boolean</td>
<td></td>
<td>When set to true the handler will be called back when a mouse button is initially pressed while the cursor is over the target object. If no target is specified the handler will call back on any button press.</td>
</tr>
<tr>
<td>onMouseHit</td>
<td>Boolean</td>
<td></td>
<td>When set to true the handler will be called back continuously when the cursor is over the target object. In the case of onMouseHit it doesn’t 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.</td>
</tr>
<tr>
<td>onMouseMove</td>
<td>Boolean</td>
<td></td>
<td>When set to true the handler will be called back when the cursor moves over the target object. If no target is specified the handler will call back on any mouse movement over the 3D annotation.</td>
</tr>
<tr>
<td>onMouseOut</td>
<td>Boolean</td>
<td></td>
<td>When set to true the handler will be called back once when the cursor moves off of the target object. In order to be called back the target must be the frontmost object. To exclude objects use <code>Node.hitEnabled</code> property.</td>
</tr>
<tr>
<td>onMouseOver</td>
<td>Boolean</td>
<td></td>
<td>When set to true the handler will be called back once when the cursor moves over the target object.</td>
</tr>
<tr>
<td>onMouseUp</td>
<td>Boolean</td>
<td></td>
<td>When set to true the handler will be called back when a mouse button is initially released. If a target is specified it will only call back when the cursor is over the handler’s target.</td>
</tr>
</tbody>
</table>
### MouseEventHandler

#### Constructor

**Syntax**  

```
new MouseEventHandler()
```

**Returns**  

A `MouseEventHandler` object

#### onEvent

A method that is called when a mouse event occurs.

**Syntax**  

```
onEvent(event)
```

**Parameters**

- `event`  
  A `MouseEvent` object representing the event.

**Returns**  

undefined

---

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
</table>
| reportAllTargets | Boolean |        | Determines whether a hit test will be performed. When set to `false`, a hit test will not be 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 `false`, the following events will not be reported because they depend on hit testing:  
  - mouse-hit  
  - mouse-move  
  - mouse-out  
  - mouse-over |
| target     | Object |        | The `Mesh` or `Background` object on which the mouse event occurs.                                                                                                                                               |
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 would print the Node object's type to the console:

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

In addition to the methods and properties below, it also contains the same methods and properties as a SceneObject, see page 75.

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>firstChild</td>
<td>Node (if the first child exists), None otherwise</td>
<td>read-only</td>
<td>The Node object's first child.</td>
</tr>
<tr>
<td>hitEnabled</td>
<td>Boolean</td>
<td></td>
<td>Determines whether the Node is included in hit tests. The default value is true.</td>
</tr>
<tr>
<td>info</td>
<td>String</td>
<td>read-only</td>
<td>Acrobat 7.0.7 Information associated with the Node.</td>
</tr>
<tr>
<td>metadataString</td>
<td>String</td>
<td>read-only</td>
<td>Acrobat 8.1 A string containing Node-specific metadata.</td>
</tr>
<tr>
<td>nextSibling</td>
<td>Node (if the next sibling exists), None otherwise</td>
<td>read-only</td>
<td>The next sibling.</td>
</tr>
<tr>
<td>parent</td>
<td>Object</td>
<td>read-only</td>
<td>The Node object's parent Node or Scene.</td>
</tr>
<tr>
<td>transform</td>
<td>Matrix4x4</td>
<td>read-only</td>
<td>The local to world transformation matrix for the Node.</td>
</tr>
<tr>
<td>wireframeColor</td>
<td>Color</td>
<td>read-only</td>
<td>The Color object used to determine the wireframe appearance.</td>
</tr>
<tr>
<td>visible</td>
<td>Boolean</td>
<td></td>
<td>Determines whether the Node object should be shown.</td>
</tr>
</tbody>
</table>
computeBoundingBox

Acrobat 7.0.7

Computes the bounds of the Node object.

Syntax
computeBoundingBox()

Returns
A BoundingBox object.

detachFromCurrentAnimation

Removes the ability of the Node object’s currently active Animation 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, see page 57.
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

Constructor that initializes the object with the identity matrix.

**Syntax**

new Quaternion()

**Returns**

A Quaternion object

Quaternion

Constructor that initializes the object with the specified rotation matrix

**Syntax**

new Quaternion(matrix)

**Parameters**

| matrix | A Matrix4x4 object representing the rotation matrix. |

**Returns**

A Quaternion object

Quaternion

Constructor that initializes the object with the specified Quaternion

**Syntax**

new Quaternion(quaternion)

**Parameters**

| quaternion | A Quaternion 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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>quaternion</code></td>
<td>A Quaternion object used to interpolate the new object.</td>
</tr>
<tr>
<td><code>a</code></td>
<td>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.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>quaternion</code></td>
<td>A Quaternion object used to interpolate the new object.</td>
</tr>
<tr>
<td><code>a</code></td>
<td>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.</td>
</tr>
</tbody>
</table>

Returns

A Quaternion object

normalize

Normalizes the Quaternion object

Syntax

`normalize()`

Returns

undefined
RenderEvent

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

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>canvas</td>
<td>Canvas</td>
<td>read-only</td>
<td>The Canvas that is the target of the RenderEvent.</td>
</tr>
<tr>
<td>canvasPixelHeight</td>
<td>integer</td>
<td>read-only</td>
<td>The height, measured in pixels, of the Canvas for which the RenderEvent is intended.</td>
</tr>
<tr>
<td>canvasPixelWidth</td>
<td>integer</td>
<td>read-only</td>
<td>The width, measured in pixels, of the Canvas for which the RenderEvent is intended.</td>
</tr>
<tr>
<td>currentTool</td>
<td>string</td>
<td>read-only</td>
<td>The name of the current tool.</td>
</tr>
</tbody>
</table>
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, page 71. It issues a callback just before each `Canvas` is rendered.

### Constructor

#### Syntax

```javascript
new RenderEventHandler()
```

#### Returns

A `RenderEventHandler` object

### onEvent

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

#### Syntax

```javascript
onEvent(event)
```

#### Parameters

<table>
<thead>
<tr>
<th>parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
</tr>
</tbody>
</table>

#### Returns

undefined
RenderOptions

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

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boundingBoxColor</td>
<td>Color</td>
<td>read-only</td>
<td>A Color object to be applied to the bounding box.</td>
</tr>
<tr>
<td>clippingPlaneColor</td>
<td>Color</td>
<td>read-only</td>
<td>A Color object to be applied to the clipping plane.</td>
</tr>
<tr>
<td>clippingPlaneIntersectionColor</td>
<td>Color</td>
<td>read-only</td>
<td>A Color object to be applied to the clipping plane intersection.</td>
</tr>
<tr>
<td>defaultAmbientColor</td>
<td>Color</td>
<td>read-only</td>
<td>A Color object to be applied to the default ambient Material.</td>
</tr>
<tr>
<td>defaultDiffuseColor</td>
<td>Color</td>
<td>read-only</td>
<td>A Color object to be applied to the default diffuse Material.</td>
</tr>
<tr>
<td>defaultEmissiveColor</td>
<td>Color</td>
<td>read-only</td>
<td>A Color object to be applied to the default emissive Material.</td>
</tr>
<tr>
<td>defaultSpecularColor</td>
<td>Color</td>
<td>read-only</td>
<td>A Color object to be applied to the default specular Material.</td>
</tr>
<tr>
<td>illustrationRenderModeFaceColor</td>
<td>Color</td>
<td>read-only</td>
<td>Acrobat 7.0.7 The color of the faces when the render mode is Illustration.</td>
</tr>
<tr>
<td>illustrationRenderModeLineColor</td>
<td>Color</td>
<td>read-only</td>
<td>A Color object to be applied to the illustration lines.</td>
</tr>
<tr>
<td>pointsRenderModeColor</td>
<td>Color</td>
<td>read-only</td>
<td>A Color object to be applied to the vertices in point render mode.</td>
</tr>
<tr>
<td>shadedIllustrationRenderModeLineColor</td>
<td>Color</td>
<td>read-only</td>
<td>A Color object to be applied to the shaded illustration lines.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Access</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| solidGridColorEven             | Color   | read-only | Acrobat 7.0.7  
The color of the even squares of the checkered grid when drawn in "solid" mode. |
| solidGridColorOdd              | Color   | read-only | Acrobat 7.0.7  
The color of the odd squares of the checkered grid when drawn in "solid" mode. |
| solidRenderModeLineColor       | Color   | read-only | A Color object to be applied to the solid or transparent lines in render mode. |
| transparentBoundsRenderModeColor | Color | read-only | A Color object to be applied to the transparent bounding box. |
| transparentGridColorEven      | Color   | read-only | Acrobat 7.0.7  
The color of the even squares of the checkered grid when drawn in "transparent" mode. |
| transparentGridColorOdd       | Color   | read-only | Acrobat 7.0.7  
The color of the odd squares of the checkered grid when drawn in "transparent" mode. |
| wireframeRenderModeColor       | Color   | read-only | Acrobat 7.0.7  
The color of the wires when the render mode is Wireframe. |
| xAxisColor                     | Color   | read-only | Acrobat 7.0.7  
The color of the x-axis. |
| yAxisColor                     | Color   | read-only | Acrobat 7.0.7  
The color of the y-axis. |
| zAxisColor                     | Color   | read-only | 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

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>string</td>
<td>read-only</td>
<td>The type of Resource object, which can be one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;image&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;model&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;unknown&quot;</td>
</tr>
<tr>
<td>TYPE_IMAGE</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the Resource type of &quot;image&quot;.</td>
</tr>
<tr>
<td>TYPE_MODEL</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the Resource type of &quot;model&quot;.</td>
</tr>
<tr>
<td>TYPE_UNKNOWN</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the Resource type of &quot;unknown&quot;.</td>
</tr>
</tbody>
</table>

Resource

Constructor

Syntax

`new Resource(pathname)`

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pathname</td>
<td>A string representing the path of the file or stream. Can only load embedded resources from within the PDF file. The pathname string must start with pdf://</td>
</tr>
</tbody>
</table>

Returns

A Resource object.
Runtime

An object that represents the runtime 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

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
</table>
| BUTTON_TYPE_PUSH    | string   | read-only | Acrobat 7.0.7  
A string constant for the custom tool button type of "push button". It is used with the addCustomToolButton method. |
| BUTTON_TYPE_TOOL    | string   | read-only | Acrobat 7.0.7  
A string constant for the custom button type of "tool button". It is used with the addCustomToolButton method. |
| canvasCount         | Number   | read-only | Acrobat 8.1  
The number of Canvases that are attached to the active 3D annotation. |
| ctrlKeyDown         | Boolean  | read-only | Determines whether the Ctrl key (Windows) or Command key (Mac OS) was pressed. |
| eventHandlerCount   | integer  | read-only | The number of registered event handlers. |
| instances           | Array    | read-only | Acrobat 7.0.7  
An array of JavaScript Annot3D objects that are attached to the 3D script context. |
| MENU_ITEM_TYPE_CHECKED | string | read-only | Acrobat 7.0.7  
A string constant for the custom menu item type of "checked". It is used with the addCustomMenuItem method. |
| MENU_ITEM_TYPE_DEFAULT | string | read-only | Acrobat 7.0.7  
A string constant for the custom menu item type of "default". It is used with the addCustomMenuItem method. |
| overrideNavTools    | Boolean  |         | Determines whether to disable all default navigation behavior. |

**Note:** Setting this property does not prevent view changes.
<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>overridePanTool</td>
<td>Boolean</td>
<td></td>
<td>Determines whether to override the built-in Pan tool behavior.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> Setting this property does not effect the pan behavior of other navigation tools.</td>
</tr>
<tr>
<td>overrideRotateTool</td>
<td>Boolean</td>
<td></td>
<td>Determines whether to override the built-in Rotate tool behavior.</td>
</tr>
<tr>
<td>overrideSelection</td>
<td>Boolean</td>
<td></td>
<td>Acrobat 7.0.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Determines whether to override the built-in Selection tool behavior.</td>
</tr>
<tr>
<td>overrideSpinTool</td>
<td>Boolean</td>
<td></td>
<td>Acrobat 8.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Determines whether to override the built-in Spin tool behavior.</td>
</tr>
<tr>
<td>overrideViewChange</td>
<td>Boolean</td>
<td></td>
<td>Determines whether to override the setting of Views from Acrobat.</td>
</tr>
<tr>
<td>overrideWalkTool</td>
<td>Boolean</td>
<td></td>
<td>Determines whether to override the built-in Walk tool behavior.</td>
</tr>
<tr>
<td>overrideWheelSpeed</td>
<td>Number</td>
<td></td>
<td>Acrobat 8.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A speed multiplier for the value of the scroll-wheel motion.</td>
</tr>
<tr>
<td>overrideZoomTool</td>
<td>Boolean</td>
<td></td>
<td>Determines whether to override the built-in Zoom tool behavior.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> Setting this property does not effect the zoom behavior of other navigation tools.</td>
</tr>
<tr>
<td>scrollWheelSpeed</td>
<td>Number</td>
<td></td>
<td>Acrobat 8.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A speed multiplier for the value of the scroll-wheel motion.</td>
</tr>
<tr>
<td>shiftKeyDown</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the Shift key was pressed.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Access</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>speedThreshold</td>
<td>Number</td>
<td>Acrobat 8.1</td>
<td>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. 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 will be too quick for very small models.</td>
</tr>
<tr>
<td>strafeSpeed</td>
<td>Number</td>
<td>Acrobat 8.1</td>
<td>A speed multiplier for the lateral motion while using the Walk tool.</td>
</tr>
<tr>
<td>TOOL_NAME_MEASURE</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the name of the measure tool. Its value is &quot;Measure&quot;.</td>
</tr>
<tr>
<td>TOOL_NAME_PAN</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the name of the pan tool. Its value is &quot;Pan&quot;.</td>
</tr>
<tr>
<td>TOOL_NAME_ROTATE</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the name of the rotate tool. Its value is &quot;Rotate&quot;.</td>
</tr>
<tr>
<td>TOOL_NAME_SPIN</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 8.0 A string constant for the name of the Spin tool. Its value is &quot;Spin&quot;.</td>
</tr>
<tr>
<td>TOOL_NAME_WALK</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the name of the walk tool. Its value is &quot;Walk&quot;.</td>
</tr>
<tr>
<td>TOOL_NAME_ZOOM</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the name of the zoom tool. Its value is &quot;Zoom&quot;.</td>
</tr>
<tr>
<td>version</td>
<td>Number</td>
<td>read-only</td>
<td>The number corresponding to the version of the Runtime system.</td>
</tr>
<tr>
<td>walkSpeed</td>
<td>Number</td>
<td>Acrobat 8.1</td>
<td>A speed multiplier for the forward/backward motion while using the Walk tool.</td>
</tr>
</tbody>
</table>
addCustomMenuItem

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

Syntax

addCustomMenuItem(name, label, type, checkedState)

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>A string identifying the menu item.</td>
</tr>
<tr>
<td>label</td>
<td>A string appearing on the menu item.</td>
</tr>
<tr>
<td>type</td>
<td>A string indicating whether it is a checked menu item. A checked menu item</td>
</tr>
<tr>
<td></td>
<td>has a check mark toggle next to it. Its possible values are:</td>
</tr>
<tr>
<td></td>
<td>- &quot;default&quot;</td>
</tr>
<tr>
<td></td>
<td>- &quot;checked&quot;</td>
</tr>
<tr>
<td>checkedState</td>
<td>A Boolean value indicating the state of a checked menu item.</td>
</tr>
</tbody>
</table>

Returns

undefined

addCustomToolButton

Creates a custom tool button in the 3D toolbar.

Syntax

addCustomToolButton(name, label, type)

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>A string identifying the tool button.</td>
</tr>
<tr>
<td>label</td>
<td>A string appearing on the tool button.</td>
</tr>
<tr>
<td>type</td>
<td>A string indicating whether it is a tool button or a push button. Its possible values are:</td>
</tr>
<tr>
<td></td>
<td>- &quot;tool button&quot;</td>
</tr>
<tr>
<td></td>
<td>- &quot;push button&quot;</td>
</tr>
</tbody>
</table>

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

refresh

Version 7.0.1

Marks the render area dirty so that it will be redrawn. This is useful when something has changed 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)
removeCustomMenuItem

Removes the custom menu item with the specified ID.

**Syntax**

```javascript
removeCustomMenuItem(menuName)
```

**Parameters**

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

**Returns**

undefined

removeCustomToolButton

Removes the custom tool button with the specified ID.

**Syntax**

```javascript
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**

```javascript
setCurrentTool(toolName)
```

**Parameters**

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

| toolName       | A string identifying the tool |

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 on page 15, Light on page 37, Material on page 39 and Mesh on page 52.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ambientIlluminationColor</td>
<td>Color</td>
<td>read-only</td>
<td>Modulates the ambient Color of all materials.</td>
</tr>
<tr>
<td>animations</td>
<td>SceneObjectList</td>
<td>read-only</td>
<td>A list of all Animation objects.</td>
</tr>
<tr>
<td>cameras</td>
<td>SceneObjectList</td>
<td>read-only</td>
<td>A list of all Camera objects in the Scene.</td>
</tr>
<tr>
<td>defaultRenderOptions</td>
<td>RenderOptions</td>
<td>read-only</td>
<td>A set of all default rendering options for the Scene.</td>
</tr>
<tr>
<td>gridMode</td>
<td>string</td>
<td></td>
<td>Acrobat 7.0.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The display style of the grid that represents a portion of the ground plane</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>in the Scene. It can have one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;off&quot; — no grid</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;wire&quot; — a wireframe grid</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;solid&quot; — a solid checkerboard grid</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;transparent&quot; — a semi-transparent checkerboard grid</td>
</tr>
<tr>
<td>GRID_MODE_OFF</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A string constant for the grid mode of &quot;off&quot;.</td>
</tr>
<tr>
<td>GRID_MODE_SOLID</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A string constant for the grid mode of &quot;solid&quot;.</td>
</tr>
<tr>
<td>GRID_MODE_TRANSPARENT</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A string constant for the grid mode of &quot;transparent&quot;.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Access</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------</td>
<td>--------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| GRID_MODE_WIRE         | string | read-only | Acrobat 7.0.7  
A string constant for the grid mode of "wire".                      |
| gridSize               | number | read-only | Acrobat 7.0.7  
The number of squares on the ground plane grid.                     |
| lengthUnits            | number | read-only | Acrobat 7.0.7  
The scale of a unit of length, specified in meters.                |
| LIGHT_MODE_FILE        | string | read-only | Acrobat 7.0.7  
A string constant for the light mode of "file".                      |
| LIGHT_MODE_NONE        | string | read-only | Acrobat 7.0.7  
A string constant for the light mode of "none".                      |
| LIGHT_MODE_WHITE       | string | read-only | Acrobat 7.0.7  
A string constant for the light mode of "white".                      |
| LIGHT_MODE_DAY         | string | read-only | Acrobat 7.0.7  
A string constant for the light mode of "day".                        |
| LIGHT_MODE_BRIGHT      | string | read-only | Acrobat 7.0.7  
A string constant for the light mode of "bright".                     |
| LIGHT_MODE_RGB         | string | read-only | Acrobat 7.0.7  
A string constant for the light mode of "rgb".                        |
| LIGHT_MODE_NIGHT       | string | read-only | Acrobat 7.0.7  
A string constant for the light mode of "night".                      |
| LIGHT_MODE_BLUE        | string | read-only | Acrobat 7.0.7  
A string constant for the light mode of "blue".                       |
| LIGHT_MODE_RED         | string | read-only | Acrobat 7.0.7  
A string constant for the light mode of "red".                        |
| LIGHT_MODE_CUBE        | string | read-only | Acrobat 7.0.7  
A string constant for the light mode of "cube".                       |
<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIGHT_MODE_CAD</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 - A string constant for the light mode of &quot;cad&quot;.</td>
</tr>
<tr>
<td>LIGHT_MODE_HEADLAMP</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 - A string constant for the light mode of &quot;headlamp&quot;.</td>
</tr>
<tr>
<td>lights</td>
<td>SceneObjectList</td>
<td>read-only</td>
<td>A list of all Light objects in the Scene.</td>
</tr>
<tr>
<td>lightScaleFactor</td>
<td>number</td>
<td></td>
<td>A uniform scale factor for all Light objects in the Scene.</td>
</tr>
<tr>
<td>lightScheme</td>
<td>string</td>
<td></td>
<td>Acrobat 7.0.7 - The current, preconfigured lighting scheme for the Scene.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>It can take one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;file&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;none&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;white&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;day&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;bright&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;rgb&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;night&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;blue&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;red&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;cube&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;cad&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● &quot;headlamp&quot;</td>
</tr>
<tr>
<td>materials</td>
<td>SceneObjectList</td>
<td>read-only</td>
<td>A list of all Material objects.</td>
</tr>
<tr>
<td>meshes</td>
<td>SceneObjectList</td>
<td>read-only</td>
<td>A list of all Mesh objects in the Scene.</td>
</tr>
<tr>
<td>nodes</td>
<td>SceneObjectList</td>
<td>read-only</td>
<td>A list of all Node objects except the default Camera and default Light objects.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Access</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>outlineAngle</td>
<td>number</td>
<td>Acrobat 7.0.7</td>
<td>The crease angle (in degrees) for the appearance of lines in Illustration render modes.</td>
</tr>
<tr>
<td>showGrid</td>
<td>Boolean</td>
<td>Acrobat 7.0.7</td>
<td>Determines whether the ground plane grid is displayed.</td>
</tr>
<tr>
<td>renderDoubleSided</td>
<td>Boolean</td>
<td>Acrobat 8.1</td>
<td>Toggles if backfacing polygons are rendered.</td>
</tr>
<tr>
<td>renderMode</td>
<td>string</td>
<td>Acrobat 7.0.7</td>
<td>The default rendering type for all objects in the Scene, which can be one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;default&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;bounding box&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;transparent bounding box&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;transparent bounding box outline&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;vertices&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;shaded vertices&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;wireframe&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;shaded wireframe&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;solid&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;transparent&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;solid wireframe&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;transparent wireframe&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;illustration&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;solid outline&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;shaded illustration&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- &quot;hidden wireframe&quot;</td>
</tr>
</tbody>
</table>

| RENDER_MODE_DEFAULT  | string       | read-only  | Acrobat 7.0.7 | A string constant for the render mode of "default".                                                                 |


<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RENDER_MODE_BOUNDING_BOX</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the render mode of &quot;bounding box&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_TRANSPARENT_BOUNDING_BOX</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the render mode of &quot;transparent bounding box&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_TRANSPARENT_BOUNDING_BOX_OUTLINE</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the render mode of &quot;transparent bounding box outline&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_VERTICES</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the render mode of &quot;vertices&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_SHADED_VERTICES</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the render mode of &quot;shaded vertices&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_WIREFRAME</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the render mode of &quot;wireframe&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_SHADED_WIREFRAME</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the render mode of &quot;shaded wireframe&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_SOLID</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the render mode of &quot;solid&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_TRANSPARENT</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the render mode of &quot;transparent&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_SHADED_SOLID_WIREFRAME</td>
<td>string</td>
<td>read-only</td>
<td>Acrobat 7.0.7 A string constant for the render mode of &quot;solid wireframe&quot;.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Access</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RENDER_MODE_TRANSPARENT_WIREFRAME</td>
<td>string</td>
<td>read-only</td>
<td>A string constant for the render mode of &quot;transparent wireframe&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_ILLUSTRATION</td>
<td>string</td>
<td>read-only</td>
<td>A string constant for the render mode of &quot;illustration&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_SOLID_OUTLINE</td>
<td>string</td>
<td>read-only</td>
<td>A string constant for the render mode of &quot;solid outline&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_SHADED_ILLUSTRATION</td>
<td>string</td>
<td>read-only</td>
<td>A string constant for the render mode of &quot;shaded illustration&quot;.</td>
</tr>
<tr>
<td>RENDER_MODE_HIDDEN_WIREFRAME</td>
<td>string</td>
<td>read-only</td>
<td>A string constant for the render mode of &quot;hidden wireframe&quot;.</td>
</tr>
<tr>
<td>selectedNode</td>
<td>Node</td>
<td></td>
<td>The currently selected Node.</td>
</tr>
<tr>
<td>showAxes</td>
<td>Boolean</td>
<td></td>
<td>Determines whether the world axes are displayed.</td>
</tr>
<tr>
<td>smoothing</td>
<td>Boolean</td>
<td></td>
<td>When true, smoothing is enabled for meshes in the scene.</td>
</tr>
<tr>
<td>smoothingAngle</td>
<td>number</td>
<td></td>
<td>The default smoothing angle (in degrees) for meshes in the scene.</td>
</tr>
<tr>
<td>smoothingOverride</td>
<td>Boolean</td>
<td></td>
<td>When set to true, overrides the smoothing values from the loaded model file.</td>
</tr>
</tbody>
</table>

activateAnimation

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

Syntax
activateAnimation(animation)

Parameters

| animation | The Animation object to be activated. |

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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sizeX</td>
<td>Model units in the x-direction used to size the Mesh.</td>
</tr>
<tr>
<td>sizeY</td>
<td>Model units in the y-direction used to size the Mesh.</td>
</tr>
<tr>
<td>name</td>
<td>(Optional) The name that will be assigned to the Mesh.</td>
</tr>
</tbody>
</table>

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 on page 75, Animation on page 15, Light on page 37, Material on page 39 and Mesh on page 52.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The SceneObject object’s name.</td>
</tr>
<tr>
<td>objectGUID</td>
<td>string</td>
<td>Deprecated A value that uniquely identifies the SceneObject in custom data. This property has a default value.</td>
</tr>
<tr>
<td>objectID</td>
<td>integer</td>
<td>An unsigned 32-bit value that uniquely identifies the SceneObject. This property can be assigned, but does not have a default value (it always returns 0).</td>
</tr>
</tbody>
</table>
SceneObjectList

A structure that contains references to several SceneObject objects.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>integer</td>
<td>read-only</td>
<td>The number of elements in the SceneObjectList.</td>
</tr>
</tbody>
</table>

**getByGUID**

Deprecated

Obtains the specified SceneObject object from the list

**Syntax**

getByGUID(guid)

**Parameters**

<table>
<thead>
<tr>
<th>guid</th>
<th>A string representing the GUID for the specified element.</th>
</tr>
</thead>
</table>

**Returns**

A SceneObject object

**getByID**

Obtains the specified SceneObject object from the list

**Syntax**

getByID(id)

**Parameters**

<table>
<thead>
<tr>
<th>id</th>
<th>An integer representing the object identifier for the specified SceneObject object.</th>
</tr>
</thead>
</table>

**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 SceneObject 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 SceneObject object. |

Returns
A SceneObject object

removeAll

Removes all the SceneObject objects from the list

Syntax
removeAll()

Returns
undefined

removeByIndex

Removes the specified SceneObject object from the list

Syntax
removeByIndex(index)
removeItem

Removes a `SceneObject` object from the list

**Syntax**

```javascript
removeItem(scene_object)
```

**Parameters**

| index       | An index to the specified element. |

**Returns**

`undefined`
ScrollWheelEvent

(Acrobat 8.1) An object that is passed as an argument to the onEvent method of the ScrollWheelHandler object, page 88. A ScrollWheelEvent object is created when the mouse scroll wheel is activated over an active 3D Canvas object.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>canvas</td>
<td>Canvas</td>
<td>read-only</td>
<td>The Canvas in which the ScrollWheelEvent took place.</td>
</tr>
<tr>
<td>canvasPixelHeight</td>
<td>integer</td>
<td>read-only</td>
<td>The height, measured in pixels, of the Canvas in which the ScrollWheelEvent took place.</td>
</tr>
<tr>
<td>canvasPixelWidth</td>
<td>integer</td>
<td>read-only</td>
<td>The width, measured in pixels, of the Canvas in which the ScrollWheelEvent took place.</td>
</tr>
<tr>
<td>ctrlKeyDown</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the Ctrl key (Windows) or Command key (Mac OS) was pressed.</td>
</tr>
<tr>
<td>currentTool</td>
<td>string</td>
<td>read-only</td>
<td>The name of the current tool.</td>
</tr>
<tr>
<td>deltaY</td>
<td>Number</td>
<td>read-only</td>
<td>The amount the scroll-wheel has been moved in the Y direction.</td>
</tr>
<tr>
<td>shiftKeyDown</td>
<td>Boolean</td>
<td>read-only</td>
<td>Determines whether the Shift key has been pressed.</td>
</tr>
</tbody>
</table>
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` on page 71.

**ScrollWheelEventHandler**

**Constructor**

**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**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>A <code>ScrollWheelEvent</code> object representing the event.</td>
</tr>
</tbody>
</table>

**Returns**

`undefined`
SelectionEvent

(Acrobat 8.1) An object that is passed as an argument to the `onEvent` method of the `SelectionEventHandler` object, page 90.

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

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>node</td>
<td>Node</td>
<td>read-only</td>
<td>The Node that is the target of the selection change.</td>
</tr>
<tr>
<td>selected</td>
<td>Boolean</td>
<td>read-only</td>
<td>The selected state of the target Node.</td>
</tr>
</tbody>
</table>
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 `addEventHandler` method.

**Constructor**

**Syntax**

```javascript
new SelectionEventHandler()
```

**Returns**

A `SelectionEventHandler` object.

**onEvent**

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

**Syntax**

```javascript
onEvent(event)
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>A <code>ScrollWheelEvent</code> object representing the event.</td>
</tr>
</tbody>
</table>

**Returns**

undefined
Texture

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

Properties

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

```javascript
setImage(image)
```

**Parameters**

<table>
<thead>
<tr>
<th>image</th>
<th>The Image to be used.</th>
</tr>
</thead>
</table>

**Returns**

undefined
**TimeEvent**

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

## Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>deltaTime</td>
<td>number</td>
<td>read-only</td>
<td>The difference between the current time and the last time.</td>
</tr>
<tr>
<td>time</td>
<td>number</td>
<td>read-only</td>
<td>The current time.</td>
</tr>
</tbody>
</table>
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.

**Constructor**

**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 `TimeEvent` object representing the event. |

**Returns**

`undefined`
ToolEvent

An object that is passed as an argument to the onEvent method of the ToolEventHandler object, see page 96.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>canvas</td>
<td>Canvas</td>
<td>read-only</td>
<td>The Canvas that is the target of the ToolEvent.</td>
</tr>
<tr>
<td>canvasPixelHeight</td>
<td>integer</td>
<td>read-only</td>
<td>The height, measured in pixels, of the Canvas for which the ToolEvent is intended.</td>
</tr>
<tr>
<td>canvasPixelWidth</td>
<td>integer</td>
<td>read-only</td>
<td>The width, measured in pixels, of the Canvas for which the ToolEvent is intended.</td>
</tr>
<tr>
<td>currentTool</td>
<td>string</td>
<td>read-only</td>
<td>The name of the current tool.</td>
</tr>
<tr>
<td>toolName</td>
<td>string</td>
<td>read-only</td>
<td>The name of the tool that was clicked.</td>
</tr>
</tbody>
</table>
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 addEventHandler` method, page 71.

ToolEventHandler

Constructor

**Syntax**

```javascript
new ToolEventHandler()
```

**Returns**

A `ToolEventHandler` object

onEvent

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

**Syntax**

```javascript
onEvent(event)
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event</td>
<td>A <code>ToolEvent</code> object representing the event.</td>
</tr>
</tbody>
</table>

**Returns**

`undefined`
Vector3

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

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>number</td>
<td></td>
<td>The x-component of the Vector3 object.</td>
</tr>
<tr>
<td>y</td>
<td>number</td>
<td></td>
<td>The y-component of the Vector3 object.</td>
</tr>
<tr>
<td>z</td>
<td>number</td>
<td></td>
<td>The z-component of the Vector3 object.</td>
</tr>
<tr>
<td>length</td>
<td>number</td>
<td>read-only</td>
<td>The length of the Vector3 object.</td>
</tr>
</tbody>
</table>

Vector3

Constructor that initializes the new object to $\langle 0.0, 0.0, 0.0 \rangle$.

Syntax

```javascript
new Vector3()
```

Returns

A Vector3 object

Vector3

Constructor that initializes the new object to the specified components.

Syntax

```javascript
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

```javascript
add(offset)
```

Parameters

| offset | The Vector3 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

```javascript
addInPlace(offset)
```

Parameters

| offset | The Vector3 object to be added to the current one |

Returns

undefined

addScaled

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

Syntax

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

Parameters

| offset | The Vector3 object to be added to the current one. |
| scale  | The scaling factor for the offset. |

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**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>The Vector3 object to be added to the current one.</td>
</tr>
<tr>
<td>scale</td>
<td>The scaling factor for the offset.</td>
</tr>
</tbody>
</table>

**Returns**

`undefined`

**blend**

Blends the current and specified Vector3 by the specified degree.

**Syntax**

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

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vec</td>
<td>The Vector3 object to be blended with the current one.</td>
</tr>
<tr>
<td>blendFactor</td>
<td>The degree of blending to be applied, which may be a value from 0.0 to 1.0.</td>
</tr>
</tbody>
</table>

**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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vec</td>
<td>The Vector3 object to be blended with the current one</td>
</tr>
<tr>
<td>blendFactor</td>
<td>The degree of blending to be applied, which may be a value from 0.0 to 1.0</td>
</tr>
</tbody>
</table>

### Returns

undefined

### cross

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

**Syntax**

cross(vec)

**Parameters**

| vec | The Vector3 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**

| vec | The Vector3 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 Vector3.

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 Vector3.

Returns
undefined

set

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

Syntax
set(vec)

Parameters

vec The Vector3 used to set the current Vector3.
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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>The x-component used to set the current Vector3.</td>
</tr>
<tr>
<td>y</td>
<td>The y-component used to set the current Vector3.</td>
</tr>
<tr>
<td>z</td>
<td>The z-component used to set the current Vector3.</td>
</tr>
</tbody>
</table>

Returns

undefined

set3

Deprecated

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

Syntax

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

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>The x-component used to set the current Vector3.</td>
</tr>
<tr>
<td>y</td>
<td>The y-component used to set the current Vector3.</td>
</tr>
<tr>
<td>z</td>
<td>The z-component used to set the current Vector3.</td>
</tr>
</tbody>
</table>

Returns

undefined

subtract

Subtracts the specified Vector3 from the current one.
Syntax
subtract (offset)

Parameters

offset  The Vector3 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 Vector3 object to be subtracted from the current one.

Returns

undefined
This chapter summarizes the new features and changes introduced in Acrobat 8.1 and earlier.

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 ScrollWheelEventHandler.

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 the additional properties: canvasCount, overrideSpinTool, scrollWheelSpeed, speedThreshold, strafeSpeed, and walkSpeed.

The Scene object has the additional properties: node and selected.

Deprecated objects or properties

The following APIs have been deprecated:

CameraEvent.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 the additional properties: `overrideSpinTool` and `TOOL_NAME_SPIN`. 
Index

3D JavaScript engine 9

A
Accessing the 3D JavaScript engine 9
activateAnimation method 81
add method 98
addCustomMenuItem method 70
addCustomToolButton method 70
addEventHandler method 71
addInPlace method 98
addModel method 81
addScaled method 98
addScaledInPlace method 99
ambientColor property 39
ambientIlluminationColor property 75
ambientTexture property 39
angle property 91
Animation object 15
  currentTime property 15
  endTime property 15
  framesPerSecond property 15
  length property 15
  startTime property 15
animations property 75
Annot3D.context3D property 9
ATTENUATION_ABC property 37
ATTENUATION_NONE property 37
attenuationA property 37
attenuationB property 37
attenuationC property 37
attenuationType property 37

B
Background object 16
  getColor method 16
  getImage method 16
  image property 16
  setColor method 16
  setImage method 17
background property 25
binding property 20, 23
BINDING_HORIZONTAL property 20
BINDING_MAX property 20
BINDING_MIN property 20
BINDING_VERTICAL property 20
blend method 99
blendInPlace method 99
Bone object 18
BoundingBox object 19
  center property 19
  max property 19
  min property 19
boundingBoxColor property 64
brightness property 37
bumpTexture property 39
BUTTON_TYPE_PUSH property 67
BUTTON_TYPE_TOOL property 67

C
Camera object
  binding property 20
  BINDER_HORIZONTAL property 20
  BINDER_MAX property 20
  BINDER_MIN property 20
  BINDER_VERTICAL property 20
far property 20
fov property 20
getDirectionFromScreen method 22
getScreenFromPosition method 21
near property 20
position property 20
positionLocal property 20
projectionType property 20
roll property 21
target property 21
targetPosition property 21
targetPositionLocal property 21
TYPE_ORTHOGONAL property 21
TYPE_PERSPECTIVE property 21
up property 21
upLocal property 21
viewPlaneSize property 21
CameraEvent object 23
  binding property 23
canvas property 23
currentTool property 23
far property 23
fov property 23
isNewCanvas property 23
near property 23
projectionType property 23
targetDistance property 23
transform property 23
viewPlaneSize property 23
CameraEventHandler method 24
CameraEventHandler object 24
  CameraEventHandler method 24
cameras property 75
Canvas object 25
  background property 25
  getCamera method 25
  setCamera method 25
canvas property 23, 34, 50, 53, 62, 87, 95
canvasCount property 67
canvasPixelHeight property 34, 53, 62, 87, 95
canvasPixelWidth property 34, 53, 62, 87, 95
center property 19
characterCode property 34
clampU property 91
clampV property 91
ClippingPlane object 26
  remove method 26
clippingPlaneColor property 64
canvas property 37
clippingPlaneIntersectionColor property 64
Color method 27
Color object 27
  Color method 27
  r, g, b properties 27
  set method 27
  set3 method 28
color property 37
computeBoundingBox method 58, 82
Console object 29
  print method 29
  println method 29
count property 84
createClippingPlane method 81
createLight method 81
createSquareMesh method 82
cross method 100
currentTool property 35, 53, 67, 87
determinate property 40
deltaTime property 93
deltaY property 87
detachFromCurrentAnimation method 58
determinant property 40
diffuseColor property 39
diffuseTexture property 39
direction property 37
directionLocal property 37
disableTool method 71
distance property 31
dot method 100
Dummy object 30
enableTool method 71
eTime property 15
eventHandlerCount property 67

F
far property 20, 23
firstChild property 57
fov property 20, 23
framesPerSecond property 15

G
getByGUID method 84
getByID method 84
getByIndex method 84
getByName method 85
getCamera method 25
getColor method 16
getDirectionFromScreen method 22
getEventHandler method 72
getImage method 16, 91
getRendererName method 72
getScreenFromPosition method 21
Getting the SceneContext3d object 9
GRID_MODE_OFF property 75
GRID_MODE_SOLID property 75
GRID_MODE_TRANSPARENT property 75
GRID_MODE_WIRE property 76
gridMode property 75
gridSize property 76

H
height property 33
hitEnabled property 57
HitInfo object 31
  material property 31
  position property 31
  surfaceNormal property 31
  target property 31
  textureCoordinate property 31
hits property 53
Host object 32

I
illustrationRenderModeFaceColor property 64
illustrationRenderModeLineColor property 64
Image method 33
Image object 33
  height property 33
  Image method 33
  width property 33
image property 16, 91
info property 57
innerConeAngle property 38
innerRadius property 38
instances property 67
interpolate method 61
interpolateInPlace method 61
emissiveColor property 39
emissiveTexture property 39
inverse property 40
invertInPlace method 40
isDoubleClick property 53
isEqual method 41
isMouseDown property 53
isMouseHit property 53
isMouseMove property 53
isMouseOut property 53
isMouseOver property 53
isMouseUp property 53
isNewCanvas property 23

K
KeyEvent object
canvas property 34
canvasPixelHeight property 34
canvasPixelWidth property 34
characterCode property 34
ctrlKeyDown property 35
currentTool property 35
shiftKeyDown property 35
KeyEventHandler method 36
KeyEventHandler object
KeyEventHandler method 36
onEvent method 36

L
leftButtonDown property 53
length property 15, 97
lengthUnits property 76
Light object 37
ATTENUATION_ABC property 37
ATTENUATION_NONE property 37
attenuationA property 37
attenuationB property 37
attenuationC property 37
attenuationType property 37
brightness property 37
color property 37
direction property 37
innerConeAngle property 38
innerRadius property 38
outerConeAngle property 38
outerRadius property 38
position property 38
positionLocal property 38
type property 38
TYPE_INFINITE property 38
TYPE_POINT property 38
TYPE_SPOT property 38
LIGHT_MODE_BLUE property 76
LIGHT_MODE_BRIGHT property 76
LIGHT_MODE_CAD property 77
LIGHT_MODE_CUBE property 76
LIGHT_MODE_DAY property 76
LIGHT_MODE_FILE property 76
LIGHT_MODE_HEADLAMP property 77
LIGHT_MODE_NIGHT property 76
LIGHT_MODE_NONE property 76
LIGHT_MODE_RED property 76
LIGHT_MODE_RGB property 76
LIGHT_MODE_WHITE property 76
lights property 77
lightScaleFactor property 77
lightScheme property 77

M
Material object 39
ambientColor property 39
ambientTexture property 39
bumpTexture property 39
diffuseColor property 39
diffuseTexture property 39
emissiveColor property 39
emissiveTexture property 39
opacity property 39
opacityTexture property 39
reflectionStrength property 39
reflectionTexture property 39
specularColor property 39
specularTexture property 39
material property 31, 52
materials property 77
Matrix4x4 method 40
Matrix4x4 object
determinant property 40
inverse property 40
invertInPlace method 40
isEqual method 41
Matrix4x4 method 40
multiply method 41
multiplyInPlace method 41
rotateAboutLine method 42
rotateAboutLineInPlace method 43
rotateAboutVector method 44
rotateAboutVectorInPlace method 44
rotateAboutX method 45
rotateAboutXInPlace method 45
rotateAboutY method 45
rotateAboutYInPlace method 45
rotateAboutZ method 45
rotateAboutZInPlace method 46
rotateWithQuaternion method 42
rotateWithQuaternionInPlace method 42
scale method 46
scaleComponent property 40
scaleInPlace method 46
set method 47
setIdentity method 47
setView method 47
transformDirection method 48
transformPosition method 48
translate method 49
translateInPlace method 49
translation property 40
transpose property 40
transposeInPlace method 49
max property 19
MENU_ITEM_TYPE_CHECKED property 67
MENU_ITEM_TYPE_DEFAULT property 67
MenuEvent object 50
canvas property 50
currentTool property 50
menuItemChecked property 50
menuItemName property 50
MenuEventHandler method 51
MenuEventHandler object
MenuEventHandler method 51
onEvent method 51
menuItemChecked property 50
menuItemName property 50
Mesh object 52
material property 52
renderMode property 52
meshes property 77
metadataString property 57
min property 19
modulate property 91
MouseEvent object 53
canvas property 53, 87
canvasPixelHeight property 53, 87
canvasPixelWidth property 53, 87
currentTool property 53, 87
hits property 53
isDoubleClick property 53
isMouseDown property 53
isMove property 53
isMouseHit property 53
isMouseMove property 53
isMouseOut property 53
isMouseOver property 53
isMouseUp property 53
leftButtonDown property 53
mouseX property 53
mouseY property 53
onEvent method 56
onEvent method 56
onMouseDoubleClick property 55
onMouseDown property 55
onMouseHit property 55
onMouseMove property 55
onMouseOut property 55
onMouseOver property 55
onMouseUp property 55
opposite method 41
parent property 57
phongExponent property 39
pointsRenderModeColor property 64
position property 20, 31, 38
positionLocal property 20, 38
print method 29
println method 29
Procedural object 59
projectionType property 20, 23

N
name property 83
near property 20, 23
nextSibling property 57
Node object 57
computeBoundingBox method 58
detachFromCurrentAnimation method 58
directionLocal property 37
firstChild property 57
hitEnabled property 57
info property 57
metadataString property 57
nextSibling property 57
parent property 57
transform property 57
visible property 57
wireframeColor property 57
node property 89
nodes property 77
normalize method 61, 100

O
objectGUID property 83
objectID property 83
onEvent method 24, 36, 51, 56, 63, 88, 90, 94, 96
onMouseDoubleClick property 55
onMouseDown property 55
onMouseHit property 55
onMouseMove property 55
onMouseOut property 55
onMouseOver property 55
onMouseUp property 55
opacityTexture property 39
outerConeAngle property 38
outerRadius property 38
outlineAngle property 78
overrideNavTools property 67
overridePanTool property 68
overrideRotateTool property 68
overrideSelection property 68
overrideSpinTool property 68
overrideViewChange property 68
overrideWalkTool property 68
overrideWheelSpeed property 68
overrideZoomTool property 68

P
parent property 57
phongExponent property 39
pointsRenderModeColor property 64
position property 20, 31, 38
positionLocal property 20, 38
print method 29
println method 29
Procedural object 59
projectionType property 20, 23
### Q
- Quaternion method 60
- Quaternion object 60
  - interpolate method 61
  - interpolateInPlace method 61
  - normalize method 61

### R
- reflectionStrength property 39
- reflectionTexture property 39
- refresh method 72
- remove method 26
- removeAll method 85
- removeByIndex method 85
- removeCustomMenuItem method 73
- removeCustomToolButton method 73
- removeEventHandler method 72
- removeItem method 86
- RENDER_MODE_BOUNDING_BOX property 79
- RENDER_MODE_DEFAULT property 78
- RENDER_MODE_HIDDEN_WIREFRAME property 80
- RENDER_MODE_ILLUSTRATION property 80
- RENDER_MODE_SHADED_ILLUSTRATION property 80
- RENDER_MODE_SHADED_SOLID_WIREFRAME property 79
- RENDER_MODE_SHADED_VERTICES property 79
- RENDER_MODE_SHADED_WIREFRAME property 79
- RENDER_MODE_SOLID property 79
- RENDER_MODE_SOLID_OUTLINE property 80
- RENDER_MODE_TRANSPARENT property 79
- RENDER_MODE_TRANSPARENT_BOUNDING_BOX property 79
- RENDER_MODE_TRANSPARENT_BOUNDING_BOX_OUTLINE property 79
- RENDER_MODE_TRANSPARENT_WIREFRAME property 80
- RENDER_MODE_VERTICES property 79
- RenderEvent object 62
  - canvas property 62
  - canvasPixelHeight property 62
  - canvasPixelWidth property 62
  - currentTool property 62
- RenderEventHandler method 63
- RenderEventHandler object 63
  - onEvent method 63
  - RenderEventHandler method 63
- renderMode property 52, 78
- RenderOptions object 64
  - boundingBoxColor property 64
  - clippingPlaneColor property 64
  - clippingPlaneIntersectionColor property 64
  - defaultAmbientColor property 64
  - defaultDiffuseColor property 64
  - defaultEmissiveColor property 64
  - defaultSpecularColor property 64
  - illustrationRenderModeFaceColor property 64
  - illustrationRenderModeLineColor property 64
  - pointsRenderModeColor property 64
  - shadedIllustrationRenderModeLineColor property 64
  - ReportAllTargets property 56
  - Resource method 66
  - Resource object 66
  - solidGridColorEven property 65
  - solidGridColorOdd property 65
  - solidRenderModeLineColor property 65
  - transparentBoundsRenderModeColor property 65
  - transparentGridColorEven property 65
  - transparentGridColorOdd property 65
  - wireframeRenderModeColor property 65
  - xAxisColor property 65
  - yAxisColor property 65
  - zAxisColor property 65
- ReportAllTargets property 56
- Resource method 66
- Resource object 66
  - type property 66
  - TYPE_IMAGE property 66
  - TYPE_MODEL property 66
  - TYPE_UNKNOWN property 66
- rightButtonDown property 54
- roll property 21
- rotateAboutLine method 42
- rotateAboutLineInPlace method 43
- rotateAboutVector method 44
- rotateAboutVectorInPlace method 44
- rotateAboutX method 45
- rotateAboutXInPlace method 45
- rotateAboutY method 46
- rotateAboutYInPlace method 46
- rotateAboutZ method 46
- rotateAboutZInPlace method 46
- rotateWithQuaternion method 47
- rotateWithQuaternionInPlace method 47
- Runtime object 67
  - addCustomMenuItem method 70
  - addCustomToolButton method 70
  - addEventHandler method 71
  - BUTTON_TYPE_PUSH property 67
  - BUTTON_TYPE_TOOL property 67
  - canvasCount property 67
  - ctrlKeyDown property 67
  - disableTool method 71
  - enableTool method 71
  - eventHandlerCount property 67
  - getEventHandler method 72
  - getRendererName method 72
  - instances property 67
  - MENU_ITEM_TYPE_CHECKED property 67
  - MENU_ITEM_TYPE_DEFAULT property 67
  - overrideNavTools property 67
  - overridePanTool property 68
  - overrideRotateTool property 68
  - overrideSelection property 68
  - overrideSpinTool property 68
  - overrideViewChange property 68
  - overrideWalkTool property 68
  - overrideWheelSpeed property 68
  - overrideZoomTool property 68
  - refresh method 72
  - removeCustomMenuItem method 73
Runtime object (Continued)
removeCustomToolButton method 73
removeEventHandler method 72
scrollWheelSpeed property 68
setCurrentTool method 73
shiftKeyDown property 69
strafeSpeed property 69
TOOL_NAME_MEASURE property 69
TOOL_NAME_PAN property 69
TOOL_NAME_ROTATE property 69
TOOL_NAME_SPIN property 69
TOOL_NAME_WALK property 69
TOOL_NAME_ZOOM property 69
version property 69
walkSpeed property 69

Scene object (Continued)
RENDER_MODE_DEFAULT property 78
RENDER_MODE_HIDDEN_WIREFRAME property 80
RENDER_MODE_ILLUSTRATION property 80
RENDER_MODE_SHADED_ILLUSTRATION property 80
RENDER_MODE_SHADED_SOLID_WIREFRAME property 79
RENDER_MODE_SHADED_VERTICES property 79
RENDER_MODE_SHADED_WIREFRAME property 79
RENDER_MODE_SOLID property 79
RENDER_MODE_SOLID_OUTLINE property 80
RENDER_MODE_TRANSPARENT property 79
RENDER_MODE_TRANSPARENT_BOUNDING_BOX property 79
RENDER_MODE_TRANSPARENT_BOUNDING_BOX_OUTLINE property 79
RENDER_MODE_TRANSPARENT_WIREFRAME property 80
RENDER_MODE_VERTICES property 79
renderDoubleSided property 78
renderMode property 78
selectedNode property 80
showAxes property 80
showGrid property 78
smoothing property 80
smoothingAngle property 80
smoothingOverride property 80
update method 82

SceneObject object 83
name property 83
objectGUID property 83
objectID property 83

SceneObjectList object 84
count property 84
getByGUID method 84
getByID method 84
getByIndex method 84
getByName method 85
removeAll method 85
removeByIndex method 85
removeItem method 86

ScrollWheelEvent object
canvas property 87
canvasPixelWidth property 87
canvasPixelHeight property 87
currentTool property 87
deltaY property 87
shiftKeyDown property 87

ScrollWheelEventHandler method 88
ScrollWheelEventHandler object
onEvent method 88
scrollWheelSpeed property 68
selected property 89
selectedNode property 80
SelectionEvent object
node property 89
selected property 89
SelectionEventHandler method 90

S
scale method 46, 101
scaleComponent property 40
scaleInPlace method 46, 101
Scene object 75
activateAnimation method 81
addModel method 81
ambientIlluminationColor property 75
animations property 75
cameras property 75
computeBoundingBox method 82
createClippingPlane method 81
createLight method 81
createSquareMesh method 82
defaultRenderOptions property 75
GRID_MODE_OFF property 75
GRID_MODE_SOLID property 75
GRID_MODE_TRANSPARENT property 75
GRID_MODE_WIRE property 76
gridMode property 75
gridSize property 76
lengthUnits property 76
LIGHT_MODE_BLUE property 76
LIGHT_MODE_BRIGHT property 76
LIGHT_MODE_CAD property 77
LIGHT_MODE_CUBE property 76
LIGHT_MODE_DAY property 76
LIGHT_MODE_FILE property 76
LIGHT_MODE_HEADLAMP property 77
LIGHT_MODE_NIGHT property 76
LIGHT_MODE_NONE property 76
LIGHT_MODE_RED property 76
LIGHT_MODE_RGB property 76
LIGHT_MODE_WHITE property 76
lights property 77
lightScaleFactor property 77
lightScheme property 77
materials property 77
meshes property 77
nodes property 77
outlineAngle property 78
RENDER_MODE_BOUNDING_BOX property 79
SelectionEventHandler object
  onEvent method 90
    SelectionEventHandler method 90
set method 27, 47, 101
set3 method 28, 102
setCamera method 25
setColor method 16
currentTool method 73
setIdentity method 47
setImage method 17, 92
setView method 47
shadedIllustrationRenderModeLineColor property 64
shiftKeyDown property 35, 54, 68, 87
showAxes property 80
showGrid property 78
smoothing property 80
smoothingAngle property 80
smoothingOverride property 80
solidGridColorEven property 65
solidGridColorOdd property 65
solidRenderModeLineColor property 65
specularColor property 39
specularStrength property 39
speedThreshold property 69
startTime property 15
strafeSpeed property 69
subtract method 102
subtractInPlace method 103
surfaceNormal property 31

T
  target property 21, 31, 56
targetDistance property 23
targetPosition property 21
targetPositionLocal property 21
Texture object 91
    amount property 91
    angle property 91
    clampU property 91
    clampV property 91
    getImage method 91
    image property 91
    modulate property 91
    setImage method 91
    uOffset property 91
    uScale property 91
    use3DSStyleMapping property 91
    vOffset property 91
    vScale property 91
textureCoordinate property 31
time property 93
TimeEvent object 93
deltaTime property 93
time property 93
TimeEventHandler method 94
TimeEventHandler object 94
  onEvent method 94
    TimeEventHandler method 94
TOOL_NAME_MEASURE property 69
TOOL_NAME_PAN property 69
TOOL_NAME_ROTATE property 69
TOOL_NAME_SPIN property 69
TOOL_NAME_WALK property 69
TOOL_NAME_ZOOM property 69
ToolEvent object 95
    canvas property 95
    canvasPixelHeight property 95
    canvasPixelWidth property 95
currentTool property 95
toolName property 95
ToolEventHandler method 96
ToolEventHandler object 96
  onEvent method 96
    ToolEventHandler method 96
toolName property 95
    transform property 23, 57
    transformDirection method 48
    transformPosition method 48
    translate method 49
    translateInPlace method 49
    translation property 40
    transparentBoundsRenderModeColor property 65
    transparentGridColorEven property 65
    transparentGridColorOdd property 65
    transpose property 40
    transposeInPlace method 49
type property 38, 66
    TYPE_IMAGE property 66
    TYPE_INFINITE property 38
    TYPE_MODEL property 66
    TYPE_ORTHOGRAPHIC property 21
    TYPE_PERSPECTIVE property 21
    TYPE_POINT property 38
    TYPE_SPOT property 38
    TYPE_UNKNOWN property 66
U
  uOffset property 91
  up property 21
  update method 82
  upLocal property 21
  uScale property 91
  use3DSStyleMapping property 91
V
  Vector3 method 97
  Vector3 object 97
    add method 98
    addInPlace method 98
    addScaled method 98
    addScaledInPlace method 99
    blend method 99
    blendInPlace method 99
    cross method 100
dot method 100
length property 97
normalize method 100
scale method 101
Vector3 object  (Continued)
  scaleInPlace method  101
  set method  101
  set3 method  102
  subtract method  102
  subtractInPlace method  103
  Vector3 method  97
  x property  97
  y property  97
version property  69
viewPlaneSize property  21, 23
visible property  57
vOffset property  91
vScale property  91

W
walkSpeed property  69

width property  33
wireframeColor property  57
wireframeRenderModeColor property  65

X
x property  97
xAxisColor property  65

Y
y property  97
yAxisColor property  65

Z
z property  97
zAxisColor property  65