

---

# NSOpenGLView Class Reference

Graphics & Animation: 3D Drawing



2007-01-26



Apple Inc.  
© 2007 Apple Inc.  
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

Apple Inc.  
1 Infinite Loop  
Cupertino, CA 95014  
408-996-1010

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

OpenGL is a registered trademark of Silicon Graphics, Inc.

Simultaneously published in the United States and Canada.

**Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.**

**IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.**

**THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.**

**Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.**

# Contents

---

## **NSOpenGLView Class Reference 5**

---

Overview	5
Tasks	6
Initializing an NSOpenGLView	6
Managing the NSOpenGLPixelFormat	6
Managing the NSOpenGLContext	6
Managing the Visible Region	6
Class Methods	7
defaultPixelFormat	7
Instance Methods	7
clearGLContext	7
initWithFrame:pixelFormat:	8
openGLContext	8
pixelFormat	9
prepareOpenGL	9
reshape	10
setOpenGLContext:	10
setPixelFormat:	11
update	11

---

## **Document Revision History 13**

---

---

## **Index 15**

---



# NSOpenGLView Class Reference

---

<b>Inherits from</b>	NSView : NSResponder : NSObject
<b>Conforms to</b>	NSAnimatablePropertyContainer (NSView) NSCoding (NSResponder) NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/AppKit.framework
<b>Availability</b>	Available in Mac OS X v10.0 and later.
<b>Companion guide</b>	Cocoa Drawing Guide
<b>Declared in</b>	NSOpenGLView.h
<b>Related sample code</b>	UIColorTracking From A View to A Movie From A View to A Picture LiveVideoMixer2 LiveVideoMixer3

## Overview

An `NSOpenGLView` object maintains an `NSOpenGLPixelFormat` and `NSOpenGLContext` object into which OpenGL calls can be rendered. The view provides methods for accessing and managing the `NSOpenGLPixelFormat` and `NSOpenGLContext` objects, as well as notifications of visible region changes.

An `NSOpenGLView` object cannot have subviews. You can, however, divide a single `NSOpenGLView` into multiple rendering areas using the `glViewPort` function.

When creating an `NSOpenGLView` object in Interface Builder, you use the inspector window to specify the pixel format attributes you want for the view. Only those attributes listed in the Interface Builder inspector are set when the view is instantiated.

**Note:** In versions of the Xcode Tools that shipped prior to Mac OS X v10.4, the Interface Builder inspector does not list any pixel format attributes for `NSOpenGLView`.

## Tasks

### Initializing an `NSOpenGLView`

- `initWithFrame:pixelFormat:` (page 8)  
Returns an `NSOpenGLView` object initialized with the specified frame rectangle and pixel format.

### Managing the `NSOpenGLPixelFormat`

- + `defaultPixelFormat` (page 7)  
Returns a default `NSOpenGLPixelFormat` object.
- `PixelFormat` (page 9)  
Returns the `NSOpenGLPixelFormat` object associated with the receiver.
- `setPixelFormat:` (page 11)  
Sets the receiver's `NSOpenGLPixelFormat` object to the specified object.

### Managing the `NSOpenGLContext`

- `prepareOpenGL` (page 9)  
Used by subclasses to initialize OpenGL state.
- `clearGLContext` (page 7)  
Releases the `NSOpenGLContext` object associated with the view.
- `openGLContext` (page 8)  
Returns the `NSOpenGLContext` object associated with the receiver.
- `setOpenGLContext:` (page 10)  
Sets the `NSOpenGLContext` object associated with the receiver.

### Managing the Visible Region

- `reshape` (page 10)  
Called by Cocoa when the view's visible rectangle or bounds change.
- `update` (page 11)  
Called by Cocoa when the view's window moves or when the view itself moves or is resized.

## Class Methods

### defaultPixelFormat

Returns a default NSOpenGLPixelFormat object.

```
+ (NSOpenGLPixelFormat *)defaultPixelFormat
```

#### Return Value

A pixel format object with no attributes set.

#### Discussion

Typically used with the initializer `initWithFrame:pixelFormat:` (page 8), this object has no attributes set.

#### Availability

Available in Mac OS X v10.0 and later.

#### See Also

- [PixelFormat](#) (page 9)
- [setPixelFormat:](#) (page 11)

#### Related Sample Code

[LiveVideoMixer](#)

[LiveVideoMixer2](#)

[LiveVideoMixer3](#)

[QTQuartzPlayer](#)

[WhackedTV](#)

#### Declared In

`NSOpenGLView.h`

## Instance Methods

### clearGLContext

Releases the NSOpenGLContext object associated with the view.

```
- (void)clearGLContext
```

#### Discussion

If necessary, this method calls the `clearDrawable` method of the context object before releasing it.

#### Availability

Available in Mac OS X v10.0 and later.

#### See Also

- [OpenGLContext](#) (page 8)
- [setOpenGLContext:](#) (page 10)

**Declared In**

NSOpenGLView.h

**initWithFrame:pixelFormat:**Returns an `NSOpenGLView` object initialized with the specified frame rectangle and pixel format.

```
- (id)initWithFrame:(NSRect)frameRect pixelFormat:(NSOpenGLPixelFormat *)format
```

**Parameters***frameRect*

The frame rectangle for the view, specified in the coordinate system of its parent view.

*format*The pixel format to use when creating the view's `NSOpenGLContext` object.**Return Value**An initialized `NSOpenGLView` object, or `nil` if the object could not be initialized.**Availability**

Available in Mac OS X v10.0 and later.

**See Also**+ [defaultPixelFormat](#) (page 7)**Related Sample Code**

Cocoa OpenGL

LiveVideoMixer2

OpenCL NBody Simulation Example

OpenGL Screensaver

VBL

**Declared In**

NSOpenGLView.h

**openGLContext**Returns the `NSOpenGLContext` object associated with the receiver.

```
- (NSOpenGLContext *)openGLContext
```

**Return Value**

The OpenGL context object of the receiver.

**Discussion**

If the receiver has no associated context object, a new `NSOpenGLContext` object is created and returned. The new object is initialized with the receiver's pixel format information.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**- [clearGLContext](#) (page 7)

- [setOpenGLContext:](#) (page 10)
- [pixelFormat](#) (page 9)

#### Related Sample Code

Cocoa OpenGL  
LiveVideoMixer2  
NURBSSurfaceVertexProg  
OpenGL Screensaver  
SurfaceVertexProgram

#### Declared In

NSOpenGLView.h

## pixelFormat

Returns the NSOpenGLPixelFormat object associated with the receiver.

- (NSOpenGLPixelFormat \*)pixelFormat

#### Return Value

The receiver's pixel format object.

#### Availability

Available in Mac OS X v10.0 and later.

#### See Also

- + [defaultPixelFormat](#) (page 7)
- [initWithFrame:pixelFormat:](#) (page 8)
- [setPixelFormat:](#) (page 11)

#### Related Sample Code

CoreImageGLTextureFBO  
DispatchFractal  
Draw Pixels  
TexturePerformanceDemo  
TextureRange

#### Declared In

NSOpenGLView.h

## prepareOpenGL

Used by subclasses to initialize OpenGL state.

- (void)prepareOpenGL

#### Discussion

This method is called only once after the OpenGL context is made the current context. Subclasses that implement this method can use it to configure the Open GL state in preparation for drawing.

**Availability**

Available in Mac OS X v10.3 and later.

**Related Sample Code**

CIColorTracking

CIVideoDemoGL

FunHouse

QTCoreVideo103

WhackedTV

**Declared In**

NSOpenGLView.h

**reshape**

Called by Cocoa when the view's visible rectangle or bounds change.

```
- (void)reshape
```

**Discussion**

Cocoa typically calls this method during scrolling and resize operations but may call it in other situations when the view's rectangles change. The default implementation does nothing. You can override this method if you need to adjust the viewport and display frustum.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

CIVideoDemoGL

DispatchLife

iChatTheater

NSOpenGL Fullscreen

VertexPerformanceTest

**Declared In**

NSOpenGLView.h

**setOpenGLContext:**

Sets the NSOpenGLContext object associated with the receiver.

```
- (void)setOpenGLContext:(NSOpenGLContext *)context
```

**Parameters**

*context*

The OpenGL context object to associate with the receiver.

**Discussion**

This method releases the current OpenGL context, if one already exists. You must also call the `setView:` method of the context object to synchronize the context with the view.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [clearGLContext](#) (page 7)
- [openGLContext](#) (page 8)

**Related Sample Code**

LiveVideoMixer

LiveVideoMixer2

LiveVideoMixer3

**Declared In**

NSOpenGLView.h

**setPixelFormat:**

Sets the receiver's NSOpenGLPixelFormat object to the specified object.

```
- (void)setPixelFormat:(NSOpenGLPixelFormat *)pixelFormat
```

**Parameters**

*pixelFormat*

The new pixel format object for the receiver.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- + [defaultPixelFormat](#) (page 7)
- [PixelFormat](#) (page 9)

**Declared In**

NSOpenGLView.h

**update**

Called by Cocoa when the view's window moves or when the view itself moves or is resized.

```
- (void)update
```

**Discussion**

The default implementation simply calls the `update` method of `NSOpenGLContext`. You can override this method to perform additional update operations on the context or if you need to add locks for multithreaded access to multiple contexts.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

CIVideoDemoGL

DispatchLife

LiveVideoMixer  
LiveVideoMixer2  
LiveVideoMixer3

**Declared In**

NSOpenGLView.h

# Document Revision History

---

This table describes the changes to *NSOpenGLView Class Reference*.

Date	Notes
2007-01-26	Updated for Mac OS X v10.5.
2006-05-23	First publication of this content as a separate document.

## REVISION HISTORY

### Document Revision History

# Index

---

## C

---

`clearGLContext` [instance method 7](#)

## D

---

`defaultPixelFormat` [class method 7](#)

## I

---

`initWithFrame:pixelFormat:` [instance method 8](#)

## O

---

`openGLContext` [instance method 8](#)

## P

---

`PixelFormat` [instance method 9](#)

`prepareOpenGL` [instance method 9](#)

## R

---

`reshape` [instance method 10](#)

## S

---

`setOpenGLContext:` [instance method 10](#)

`setPixelFormat:` [instance method 11](#)

## U

---

`update` [instance method 11](#)