
NSOpenGLPixelFormat Class Reference

Graphics & Animation: 3D Drawing





Apple Inc.
© 2009 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 Quartz 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

NSOpenGLPixelFormat Class Reference 5

Overview 5

Tasks 5

 Initializing an OpenGL Pixel Buffer 5

 Obtaining Information About an OpenGL Pixel Buffer 6

Instance Methods 6

 CGLPBufferObj 6

 initWithCGLPBufferObj: 6

 initWithTextureTarget:textureInternalFormat:textureMaxMipMapLevel:pixelsWide: pixelsHigh:
7

 pixelsHigh 8

 pixelsWide 8

 textureInternalFormat 9

 textureMaxMipMapLevel 9

 textureTarget 9

Document Revision History 11

Index 13

NSOpenGLPixelFormat Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/AppKit.framework
Availability	Available in Mac OS X v10.3 and later.
Companion guide	Cocoa Drawing Guide
Declared in	NSOpenGL.h
Related sample code	Quartz Composer Offline Rendering Quartz Composer Texture

Overview

The `NSOpenGLPixelFormat` class gives Cocoa OpenGL implementations access to accelerated offscreen rendering. With this offscreen rendering you could, for instance, draw into the pixel buffer, then use the contents as a texture map elsewhere. Typically you initialize an `NSOpenGLPixelFormat` object using the `initWithTextureTarget:textureInternalFormat:textureMaxMipMapLevel:pixelWide:pixelHigh:` (page 7) method and attach the resulting object to an OpenGL context with the `setPixelFormat:cubeMapFace:mipMapLevel:currentVirtualScreen:` method of `NSOpenGLContext`.

Every `NSOpenGLPixelFormat` object wraps a low-level, platform-specific Core OpenGL (CGL) pixel buffer object. Your application can retrieve the CGL pixel buffer by calling the `CGLBufferObj` (page 6) method. For more information on the underlying CGL pixel buffer, see *CGL Reference*.

Tasks

Initializing an OpenGL Pixel Buffer

- `initWithTextureTarget:textureInternalFormat:textureMaxMipMapLevel:pixelWide:pixelHigh:` (page 7)
Returns an `NSOpenGLPixelFormat` object initialized with the specified parameters.
- `initWithCGLBufferObj:` (page 6)
Initializes and returns an `NSOpenGLPixelFormat` object that encapsulates an existing CGL pixel buffer object.

Obtaining Information About an OpenGL Pixel Buffer

- [CGLPBufferObj](#) (page 6)
Returns the underlying `CGLPBufferObj` object associated with the `NSOpenGLPixelFormat` object.
- [pixelsHigh](#) (page 8)
Returns the height of the receiver's texture (in pixels).
- [pixelsWide](#) (page 8)
Returns the width of the receiver's texture (in pixels).
- [textureInternalFormat](#) (page 9)
Returns the internal format of the receiver's texture.
- [textureMaxMipMapLevel](#) (page 9)
Returns the maximum mipmap level of the receiver's texture.
- [textureTarget](#) (page 9)
Returns the texture target of the receiver.

Instance Methods

CGLPBufferObj

Returns the underlying `CGLPBufferObj` object associated with the `NSOpenGLPixelFormat` object.

```
- (void *)CGLPBufferObj
```

Return Value

The CGL pixel buffer object that encapsulates the actual pixel buffer.

Availability

Available in Mac OS X v10.6 and later.

Declared In

`NSOpenGL.h`

initWithCGLPBufferObj:

Initializes and returns an `NSOpenGLPixelFormat` object that encapsulates an existing CGL pixel buffer object.

```
- (id)initWithCGLPBufferObj:(void *)pbuffer
```

Parameters

pbuffer

The CGL pixel buffer object to wrap.

Return Value

An initialized `NSOpenGLPixelFormat` object.

Discussion

If your application already has a CGL pixel buffer object, you can wrap it inside an `NSOpenGLPixelFormat` object by using this initializer. This method retains the CGL pixel buffer object by calling the `CGLRetainPBuffer` function.

Your application should not call `CGLDestroyPBuffer` to dispose of the CGL pixel buffer object. Instead, your application should call `CGLReleasePBuffer` to decrement its reference count.

Availability

Available in Mac OS X v10.6 and later.

Declared In

`NSOpenGL.h`

`initWithTextureTarget:textureInternalFormat:textureMaxMipMapLevel:pixelsWide:pixelsHigh:`

Returns an `NSOpenGLPixelFormat` object initialized with the specified parameters.

```
- (id)initWithTextureTarget:(GLenum)target textureInternalFormat:(GLenum)format
 textureMaxMipMapLevel:(GLint)maxLevel pixelsWide:(GLsizei)pixelsWide
 pixelsHigh:(GLsizei)pixelsHigh
```

Parameters

target

The texture object. This value should be one of the following:

`GL_TEXTURE_2D`, `GL_TEXTURE_CUBE_MAP`, or `GL_TEXTURE_RECTANGLE_EXT`.

format

The base internal format of the texture. This value should be `GL_RGB`, `GL_RGBA`, or `GL_DEPTH_COMPONENT`.

maxLevel

The desired maximum mipmap level of the structure, starting with zero.

pixelsWide

The width of the texture (in pixels) in the pixel buffer.

pixelsHigh

The height of the texture (in pixels) in the pixel buffer.

Return Value

An initialized `NSOpenGLPixelFormat` object or `nil` if the initialization failed. Initialization can fail if there is inconsistency among the parameter values. See the OpenGL documentation for `glTexImage2D` for more information.

Discussion

The value you pass to the *target* parameter defines several other constraints that are then applied to the remaining parameters. The list below gives the values you can pass to *target* and the additional constraints.

- `GL_TEXTURE_2D`
- `GL_TEXTURE_CUBE_MAP` - the values in *pixelsWide* and *pixelsHigh* must be equal.
- `GL_TEXTURE_RECTANGLE_EXT` - *maxLevel* must be zero.

Normally, when using the `GL_TEXTURE_2D` and `GL_TEXTURE_CUBE_MAP` targets, you must specify width and height values that are powers of two. When the `ARB_texture_non_power_of_two` extension is present, however, some types of hardware can support values that are not powers of two. You should check for the presence of this extension before specifying non power-of-two values.

If the texture map cannot be created, you can use the `glGetError` function to get the error code.

Availability

Available in Mac OS X v10.3 and later.

Related Sample Code

Quartz Composer Offline Rendering

Quartz Composer Texture

Declared In

`NSOpenGL.h`

pixelsHigh

Returns the height of the receiver's texture (in pixels).

- (`GLsizei`)pixelsHigh

Return Value

The height of the texture (in pixels).

Availability

Available in Mac OS X v10.3 and later.

See Also

- [pixelsWide](#) (page 8)

Related Sample Code

Quartz Composer Offline Rendering

Quartz Composer Texture

Declared In

`NSOpenGL.h`

pixelsWide

Returns the width of the receiver's texture (in pixels).

- (`GLsizei`)pixelsWide

Return Value

The width of the texture (in pixels).

Availability

Available in Mac OS X v10.3 and later.

See Also

- [pixelsHigh](#) (page 8)

Related Sample Code

Quartz Composer Offline Rendering

Quartz Composer Texture

Declared In

NSOpenGL.h

textureInternalFormat

Returns the internal format of the receiver's texture.

- (GLenum)textureInternalFormat

Return Value

The texture format, which can be one of the following values: `GL_RGB`, `GL_RGBA`, or `GL_DEPTH_COMPONENT`.

Availability

Available in Mac OS X v10.3 and later.

Declared In

NSOpenGL.h

textureMaxMipMapLevel

Returns the maximum mipmap level of the receiver's texture.

- (GLint)textureMaxMipMapLevel

Return Value

The maximum mipmap level.

Availability

Available in Mac OS X v10.3 and later.

Declared In

NSOpenGL.h

textureTarget

Returns the texture target of the receiver.

- (GLenum)textureTarget

Return Value

The texture target, which can be one of the following values: `GL_TEXTURE_2D`, `GL_TEXTURE_CUBE_MAP`, or `GL_TEXTURE_RECTANGLE_EXT`.

Availability

Available in Mac OS X v10.3 and later.

Declared In

NSOpenGL.h

Document Revision History

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

Date	Notes
2009-08-13	Added support for new Mac OS X v10.6 symbols.
2007-01-31	Removed references to 1D pBuffer targets, which are not supported. Corrected information about power-of-two width and height values. 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

CGLPBufferObj **instance method** [6](#)

I

initWithCGLPBufferObj: **instance method** [6](#)
initWithTextureTarget:textureInternalFormat:
textureMaxMipMapLevel:pixelsWide:pixelsHigh:
instance method [7](#)

P

pixelsHigh **instance method** [8](#)
pixelsWide **instance method** [8](#)

T

textureInternalFormat **instance method** [9](#)
textureMaxMipMapLevel **instance method** [9](#)
textureTarget **instance method** [9](#)