

---

# IMVideoDataSource Protocol Reference

[Cocoa](#) > [Apple Applications](#)



2007-07-08



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, iChat, Mac, and Mac OS 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

---

## **IMVideoDataSource Protocol Reference 5**

---

Overview 5

Tasks 5

    Providing Pixel Buffered Video 5

    Providing OpenGL Buffered Video 5

Instance Methods 6

    getOpenGLBufferContext:pixelFormat: 6

    getPixelBufferPixelFormat: 6

    renderIntoOpenGLBuffer:onScreen:forTime: 7

    renderIntoPixelBuffer:forTime: 7

---

## **Document Revision History 9**

---

**Index 11**

---



# IMVideoDataSource Protocol Reference

(informal protocol)

---

<b>Framework</b>	/System/Library/Frameworks/InstantMessage.framework
<b>Companion guide</b>	Instant Message Programming Guide
<b>Declared in</b>	IMAVManager.h

## Overview

`IMVideoDataSource` is an informal protocol that an `IMAVManager` data source must conform to in order to provide video data to iChat AV.

To provide video when the `CVPixelFormat` representation is preferred, the data source must implement both the [getPixelFormatPixelFormat:](#) (page 6) and [renderIntoPixelFormat:forTime:](#) (page 7) methods. Otherwise, to provide video when the `CVOpenGLBuffers` representation is preferred, the data source must implement both the [getOpenGLBufferContext:pixelFormat:](#) (page 6) and [renderIntoOpenGLBuffer:onScreen:forTime:](#) (page 7) methods.

## Tasks

### Providing Pixel Buffered Video

- [getPixelFormatPixelFormat:](#) (page 6)  
Returns the pixel buffer format.
- [renderIntoPixelFormat:forTime:](#) (page 7)  
Provides data for the next video frame using pixel buffering.

### Providing OpenGL Buffered Video

- [getOpenGLBufferContext:pixelFormat:](#) (page 6)  
Returns the pixel OpenGL buffer context and pixel format.
- [renderIntoOpenGLBuffer:onScreen:forTime:](#) (page 7)  
Provides data for the next video frame using OpenGL buffering.

## Instance Methods

### getOpenGLBufferContext:pixelFormat:

Returns the pixel OpenGL buffer context and pixel format.

```
- (void)getOpenGLBufferContext:(CGLContextObj *)contextOut
  pixelFormat:(CGLPixelFormatObj *)pixelFormatOut
```

#### Parameters

*contextOut*

The OpenGL context to be used for the `CVOpenGLBufferRef` instances passed to the `renderIntoOpenGLBuffer:onScreen:forTime:` method.

*pixelFormatOut*

The OpenGL pixel format to be used for the `CVOpenGLBufferRef` instances passed to the `renderIntoOpenGLBuffer:onScreen:forTime:` method.

#### Discussion

This method is invoked once after `setVideoDataSource:` is sent to an `IMAVManager` object.

#### Availability

Available in Mac OS X v10.5 and later.

#### See Also

- [renderIntoOpenGLBuffer:onScreen:forTime:](#) (page 7)

#### Declared In

`IMAVManager.h`

### getPixelBufferPixelFormat:

Returns the pixel buffer format.

```
- (void)getPixelBufferPixelFormat:(OSType *)pixelFormatOut
```

#### Parameters

*pixelFormatOut*

The pixel format to be used for the `CVPixelFormatRef` instances passed to the `renderIntoPixelFormat:forTime:` method.

#### Discussion

This method is invoked once after `setVideoDataSource:` is sent to an `IMAVManager` object.

#### Availability

Available in Mac OS X v10.5 and later.

#### See Also

- [renderIntoPixelFormat:forTime:](#) (page 7)

#### Declared In

`IMAVManager.h`

## renderIntoOpenGLBuffer:onScreen:forTime:

Provides data for the next video frame using OpenGL buffering.

```
- (BOOL)renderIntoOpenGLBuffer:(CVOpenGLBufferRef)buffer onScreen:(int *)screenInOut
    forTime:(CVTimeStamp *)timeStamp
```

### Parameters

*buffer*

The OpenGL buffer to fill. The receiver should call the `CVOpenGLBufferAttach` function and then fill the buffer with video data.

*screenInOut*

The recommended virtual screen number to pass to the `CVOpenGLBufferAttach` function for maximum efficiency. The receiver may use a different screen number, but it must write that value back into *screenInOut* before returning.

*timeStamp*

The frame time for which the buffer should be rendered.

You should render a video frame that corresponds to the supplied host time, `timeStamp->hostTime`, and before returning from this method, change the host time to the earliest time for which the rendered video is valid. For example, if the content is a movie, then set the host time to correspond to the rendered frame—typically, slightly earlier than the original host time. If the content is a photo slideshow, then set the host time to the time the image first appeared which can be several seconds before the original host time. Adjusting the time this way helps synchronize the audio with the video track.

### Return Value

Returns YES if the buffer is successfully filled with new frame data. Returns NO if nothing changed or an error was encountered.

### Discussion

This method is invoked each time a frame is sent to iChat AV. This method is not invoked on the main thread.

### Availability

Available in Mac OS X v10.5 and later.

### See Also

- [getOpenGLBufferContext:pixelFormat:](#) (page 6)

### Declared In

IMAVManager.h

## renderIntoPixelBuffer:forTime:

Provides data for the next video frame using pixel buffering.

```
- (BOOL)renderIntoPixelBuffer:(CVPixelBufferRef)buffer forTime:(CVTimeStamp
    *)timeStamp
```

### Parameters

*buffer*

The pixel buffer to fill with video data. The dimensions can vary. Use the `CVPixelBufferGetWidth` and `CVPixelBufferGetHeight` functions to get the dimensions each time this method is invoked.

*timeStamp*

The frame time for which the buffer should be rendered.

You should render a video frame that corresponds to the supplied host time, `timeStamp->hostTime`, and before returning from this method, change the host time to the earliest time for which the rendered video is valid. For example, if the content is a movie, then set the host time to correspond to the rendered frame—typically, slightly earlier than the original host time. If the content is a photo slideshow, then set the host time to the time the image first appeared which can be several seconds before the original host time. Adjusting the time this way helps synchronize the audio with the video track.

**Return Value**

Returns YES if the buffer is successfully filled with new frame data. Returns NO if nothing changed or an error was encountered.

**Discussion**

This method is invoked each time a frame is sent to iChat AV. This method is not invoked on the main thread.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- [getPixelFormatFormat:](#) (page 6)

**Declared In**

IMAVManager.h



# Document Revision History

---

This table describes the changes to *IMVideoDataSource Protocol Reference*.

Date	Notes
2007-07-08	New document that describes the informal protocol of an IMAVManager data source object--describes the methods you implement to supply audio and/or video to iChat AV.

## REVISION HISTORY

### Document Revision History

# Index

---

## G

---

getOpenGLBufferContext:pixelFormat: <NSObject>  
instance method [6](#)

getPixelFormat: <NSObject> instance  
method [6](#)

## R

---

renderIntoOpenGLBuffer:onScreen:forTime:  
<NSObject> instance method [7](#)

renderIntoPixelFormat:forTime: <NSObject>  
instance method [7](#)