
Quick Look Framework Reference

Data Management: File Management



2007-04-13



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, Mac, Mac OS, Objective-C, Quartz, and Xcode are trademarks of Apple Inc., registered in the United States and other countries.

Finder and Spotlight are trademarks of Apple 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

Introduction **Introduction** 5

Part I **Opaque Types** 7

Chapter 1 **QLPreviewRequest Reference** 9

Overview 9
Functions by Task 10
Functions 10
Callbacks 17
Data Types 19
Constants 19

Chapter 2 **QLThumbnailRequest Reference** 23

Overview 23
Functions by Task 24
Functions 25
Callbacks 32
Data Types 34
Constants 34

Document Revision History 37

Index 39

Introduction

Framework	/System/Library/Frameworks/QuickLook.framework
Header file directories	QuickLook.framework/Headers
Companion guide	Quick Look Programming Guide
Declared in	QLGenerator.h QLThumbnailImage.h

This collection of documents describes the functions, types, and constants of the Quick Look framework. The Quick Look framework currently contains a public header file (`QLGenerator.h`) that declares the interface for generators. Quick Look generators are plug-ins that convert a document of an uncommon or private content type to a type that Quick Look can display. Upon request, they return thumbnail images or full-scale previews of documents. Generators are based on the CFPlugIn architecture. The two opaque types declared in `QLGenerator.h` are [QLPreviewRequestRef](#) (page 19) and [QLThumbnailRequestRef](#) (page 34).

The Quick Look framework can be used to implement generator plug-ins on Mac OS X v10.5 and later.

Opaque Types

QLPreviewRequest Reference

Derived From:	CType
Framework:	QuickLook/QuickLook.h
Declared in	QLGenerator.h
Companion guide	Quick Look Programming Guide

Overview

Quick Look generators use the `QLPreviewRequest` API to help create a document preview that is returned to Quick Look clients (such as Finder and Spotlight) for display. An object of the `QLPreviewRequest` opaque type represents a request from a client for a preview of a document and is used to return the preview image the generator creates or locates.

A generator must implement the [GeneratePreviewForURL](#) (page 18) callback function to create and return a preview image requested for a given document. `QLPreviewRequest` gives you four ways to create or locate a preview for a document:

- You can create a graphics context to draw the preview in. Graphics contexts for previews are of three kinds: bitmap, single-page vector, and multi-page vector.
 - For bitmap and single-page vector contexts, use the [QLPreviewRequestCreateContext](#) (page 12) function.
 - For multi-page (PDF) contexts, use the [QLPreviewRequestCreatePDFContext](#) (page 13) function.
- The generator can extract the preview data from the document and return it to the client using the [QLPreviewRequestSetDataRepresentation](#) (page 16) function. (For this approach to work the application's document must save the preview as part of the document.) The generator can also dynamically generate the preview and return it with this function.
- As a refinement of this same approach, the generator can dynamically generate the preview as HTML data that references attachments. It can then call `QLPreviewRequestSetDataRepresentation`, passing in a content type of HTML and dictionaries specifying the attachments.

Most of the other functions of `QLPreviewRequest` let you get the data associated with the preview request, such as the URL locating the document and the UTI identifying the content type of the document.

Functions by Task

Assigning content to the preview request

[QLPreviewRequestCreateContext](#) (page 12)

Creates a graphics context to draw the preview in.

[QLPreviewRequestCreatePDFContext](#) (page 13)

Creates a PDF context suitable to draw a multi-page preview.

[QLPreviewRequestSetDataRepresentation](#) (page 16)

Sets the preview request to data saved within the document or to dynamically generated data.

[QLPreviewRequestFlushContext](#) (page 14)

Flush the context and sets the preview response.

Getting attributes of the preview request

[QLPreviewRequestCopyContentUTI](#) (page 10)

Returns the UTI for the preview request.

[QLPreviewRequestCopyOptions](#) (page 11)

Returns the options specified for the preview request.

[QLPreviewRequestCopyURL](#) (page 12)

Returns the URL of the document for which a preview is requested.

[QLPreviewRequestGetGeneratorBundle](#) (page 15)

Get the bundle of the generator receiving the preview request.

[QLPreviewRequestIsCancelled](#) (page 15)

Returns whether the preview request has been cancelled by the client.

Getting the QLPreviewRequest Type ID

[QLPreviewRequestGetTypeID](#) (page 15)

Gets the type identifier for the `QLPreviewRequest` opaque type.

Functions

QLPreviewRequestCopyContentUTI

Returns the UTI for the preview request.

```
QL_EXPORT CFStringRef QLPreviewRequestCopyContentUTI(
    QLPreviewRequestRef preview
);
```

Parameters*preview*

The preview request object.

Return Value

The UTI of the associated preview request; returns `NULL` if the UTI is not available. You should explicitly release this object when it is no longer needed.

Special Considerations

Thread-safety: This function should be called in the same thread as the preview request is made in; generally, this is the same thread in which the [GeneratePreviewForURL](#) (page 18) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLPreviewRequestCopyOptions

Returns the options specified for the preview request.

```
QL_EXPORT CFDictionaryRef QLPreviewRequestCopyOptions(
    QLPreviewRequestRef preview
);
```

Parameters*preview*

A preview request object.

Return Value

A dictionary containing the properties specified for the preview request. See [“Preview Properties”](#) (page 19) for supported options. You should explicitly release this object when it is no longer needed.

Discussion

The client sets options in the preview request to give hints to the generator. (For Mac OS X v10.5 no options are supported.)

Special Considerations

Thread-safety: This function should be called in the same thread as the preview request is made in; generally, this is the same thread in which the [GeneratePreviewForURL](#) (page 18) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLPreviewRequestCopyURL

Returns the URL of the document for which a preview is requested.

```
QL_EXPORT CFURLRef QLPreviewRequestCopyURL(
    QLPreviewRequestRef preview
);
```

Parameters

preview

The preview request object.

Return Value

The URL identifying the file for which the preview is requested.

Special Considerations

This function is thread safe.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLPreviewRequestCreateContext

Creates a graphics context to draw the preview in.

```
QL_EXPORT CGContextRef QLPreviewRequestCreateContext(
    QLPreviewRequestRef preview,
    CGSize size,
    Boolean isBitmap,
    CFDictionaryRef properties
);
```

Parameters

preview

The preview request object.

size

The size of the preview; if *isBitmap* is true the size is in pixels, otherwise it is in points.

isBitmap

true if the preview uses a bitmap-based graphics context, false otherwise. This value of this parameter affects the interpretation of the *size* parameter.

properties

A dictionary containing properties for the preview response. [“Preview Properties”](#) (page 19) lists the current property keys and describes their values.

Return Value

A Core Graphics graphics-context object that you can draw your preview image in. You should explicitly release this object when it is no longer needed.

Discussion

You can directly draw your preview data in the graphics-context object created by this function. After calling this function, you should flush the context with [QLPreviewRequestFlushContext](#) (page 14). Also be sure to release the `CGContext` object.

Quick Look provides three types of graphics contexts for drawing previews: bitmap, single-page vector-based, and multi-page vector-based (for PDF previews). You use this function to acquire a context for bitmap and single-page vector drawing; the `isBitmap` parameter is used to distinguish between them. For multi-page contexts, use the [QLPreviewRequestCreatePDFContext](#) (page 13) function.

If you prefer to work in Objective-C code, you can convert the created `CGContextRef` to a `NSGraphicsContext` object using `graphicsContextWithGraphicsPort:flipped:`.

Special Considerations

Thread-safety: This function should be called in the same thread as the preview request is made in; generally, this is the same thread in which the [GeneratePreviewForURL](#) (page 18) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Related Sample Code

QuickLookSketch

Declared In

QLGenerator.h

QLPreviewRequestCreatePDFContext

Creates a PDF context suitable to draw a multi-page preview.

```
QL_EXPORT CGContextRef QLPreviewRequestCreatePDFContext(
    QLPreviewRequestRef preview,
    const CGRect * mediaBox,
    CFDictionaryRef auxiliaryInfo,
    CFDictionaryRef properties);
);
```

Parameters

preview

The preview request object.

mediaBox

A pointer to the media box of the context.

A media box is a rectangle that defines the size and location of the PDF page. The origin of the rectangle should typically be (0,0). If you pass NULL, Quartz uses a default page size of 8.5 by 11 inches (612 by 792 points). For information see the description for `CGPDFContextCreate`.

auxiliaryInfo

A dictionary containing PDF auxiliary information. See the description of the auxiliary dictionary keys in *CGPDFContext Reference* for more information about the keys and values of this dictionary.

properties

A dictionary containing additional properties for the preview response. For information on acceptable keys and values, see [“Preview Properties”](#) (page 19).

Return Value

A reference to a Core Graphics context object that is used to display a PDF version of the preview. You should explicitly release this object when it is no longer needed.

Discussion

Be sure to bracket each PDF page written to the context with `CGPDFContextBeginPage` and `CGPDFContextEndPage` calls. After calling this function, you should flush the context with [QLPreviewRequestFlushContext](#) (page 14).

Special Considerations

Thread-safety: This function should be called in the same thread as the preview request is made in; generally, this is the same thread in which the [GeneratePreviewForURL](#) (page 18) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

`QLGenerator.h`

QLPreviewRequestFlushContext

Flush the context and sets the preview response.

```
QL_EXPORT void QLPreviewRequestFlushContext(  
    QLPreviewRequestRef preview,  
    CGContextRef context  
);
```

Parameters

preview

The preview request object.

context

The graphics context to flush.

Discussion

You should call this method immediately after drawing in the graphics contexts created by [QLPreviewRequestCreateContext](#) (page 12) and [QLPreviewRequestCreatePDFContext](#) (page 13).

Special Considerations

Thread-safety: This function should be called in the same thread as the preview request is made in; generally, this is the same thread in which the [GeneratePreviewForURL](#) (page 18) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Related Sample Code

`QuickLookSketch`

Declared In

`QLGenerator.h`

QLPreviewRequestGetGeneratorBundle

Get the bundle of the generator receiving the preview request.

```
QL_EXPORT CFBundleRef QLPreviewRequestGetGeneratorBundle(  
    QLPreviewRequestRef preview  
);
```

Parameters

preview

The preview request object.

Return Value

A reference to a bundle object representing the generator's bundle.

Special Considerations

Thread-safety: This function should be called in the same thread as the preview request is made in; generally, this is the same thread in which the [GeneratePreviewForURL](#) (page 18) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLPreviewRequestGetTypeID

Gets the type identifier for the `QLPreviewRequest` opaque type.

```
CTypeID QLPreviewRequestGetTypeID();
```

Return Value

The type identifier for the `QLPreviewRequest` opaque type.

Special Considerations

Thread-safety: This function should be called in the same thread as the preview request is made in; generally, this is the same thread in which the [GeneratePreviewForURL](#) (page 18) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLPreviewRequestIsCancelled

Returns whether the preview request has been cancelled by the client.

```

QL_EXPORT Boolean QLPreviewRequestIsCancelled(
    QLPreviewRequestRef preview
);

```

Parameters*preview*

The object representing the preview request.

Return Value

true if the request is being canceled, false otherwise.

Discussion

While computing the response, the generator can poll the preview request object with this function to determine if the client has cancelled the request. Alternatively, the generator can implement the [CancelPreviewGeneration](#) (page 17) callback. but since that function is called in a secondary thread, it is generally safer to take the polling approach.

Special Considerations

Thread-safety: This function should be called in the same thread as the preview request is made in; generally, this is the same thread in which the [GeneratePreviewForURL](#) (page 18) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLPreviewRequestSetDataRepresentation

Sets the preview request to data saved within the document or to dynamically generated data.

```

QL_EXPORT void QLPreviewRequestSetDataRepresentation(
    QLPreviewRequest preview,
    CFDataRef data,
    CFStringRef contentTypeUTI,
    CFDictionary properties
);

```

Parameters*preview*

The preview request object.

data

The data of the preview returned to the client. .

contentTypeUTI

The UTI specifying the content type of the preview.

properties

Additional properties for the preview response. For more on supported keys and values for this dictionary, see [“Preview Properties”](#) (page 19). If the saved data is HTML, you may specify a special set of properties; see the discussion below for more information.

Discussion

This function returns preview data to the client. The data is either extracted from a document (where the document's application has saved it,) or it is dynamically generated. How Quick Look handles the data depends upon the value of *contentTypeUTI*. The content data of the preview must be of a native Quick Look type. Currently supported UTIs for these types are: `kUTTypeImage`, `kUTTypePDF`, `kUTTypeHTML`, `kUTTypeXML`, `kUTTypePlainText`, `kUTTypeRTF`, `kUTTypeMovie`, and `kUTTypeAudio`.

If the UTI type is `kUTTypeHTML`, you can have the WebKit handle the layout and display of your preview. You must provide the HTML in *data* plus any attachments (for example, Address Book cards, Mail messages, or Omni Outliner documents) in the *properties* dictionary. This dictionary takes `kQLPreviewPropertyAttachmentsKey` (page 21) as its key and consists of one or more subdictionaries (one per attachment). Each subdictionary uses an arbitrary string identifier as a key; the attachment should be referenced within the HTML data using the `kQLPreviewContentIDScheme` URL scheme ("cid") and the identifier as the URL resource specifier—for example, "cid:the_identifier". The keys of the subdictionary properties are `kQLPreviewPropertyMIMETYPEKey` (page 20), `kQLPreviewPropertyTextEncodingNameKey` (page 20), and `kQLPreviewPropertyAttachmentDataKey` (page 20).

Special Considerations

Thread-safety: This function should be called in the same thread as the preview request is made in; generally, this is the same thread in which the `GeneratePreviewForURL` (page 18) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

`QLGenerator.h`

Callbacks

CancelPreviewGeneration

Defines a pointer to a function that Quick Look calls to cancel a preview request.

```
void (*CancelPreviewGeneration)(
    void *thisInterface,
    QLPreviewRequestRef preview
);
```

Parameters

thisInterface

The `CFPlugIn` COM-style interface for the generator.

preview

The object representing the preview request.

Discussion

If the client application indicates (usually because of user choice) that it is no longer interested in the preview currently being created by a Quick Look generator, this callback function is invoked. The generator can implement this function to stop creating the preview and clean up the resources used for creating it.

An alternative to implementing this callback is to periodically poll the [QLPreviewRequestRef](#) (page 19) object with [QLPreviewRequestIsCancelled](#) (page 15) to see if the request has been canceled.

Important: Because this function is called on a thread different from the one used to request that the preview be created, you should be extra careful about thread safety. If you have any doubts about thread safety, do not implement this callback.

Special Considerations

Thread-safety: For a discussions of issues and possible approaches, see [Overview of Generator Implementation](#) in *Quick Look Programming Guide*.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QuickLook/QLGenerator.h

GeneratePreviewForURL

Defines a pointer to the callback function that Quick Look calls to request a preview from a generator.

```
OSStatus (*GeneratePreviewForURL)(
    void *thisInterface,
    QLPreviewRequestRef preview,
    CFURLRef url,
    CFStringRef contentTypeUTI,
    CFDictionaryRef options
);
```

The Xcode template for Quick Look generators automatically creates a “skeletal” function with the same name as the callback symbol name: `GeneratePreviewForURL`.

Parameters

thisInterface

The `CFPlugIn` COM-style interface for the generator.

preview

The object containing all information relevant to the preview request. The generator’s role is to assign a document preview (in the requested native type) to this object before it returns.

url

A URL (represented by a `CFURLRef` object) that locates the document for which a preview is requested.

contentTypeUTI

The UTI specifying the content type of the document for which the preview is requested.

options

A dictionary of options for processing the preview. Options may be passed from the client (for example, Finder or Spotlight).

Return Value

A status code representing the result of the request. For the current version of this callback, you should always return `noErr`.

Discussion

The `GeneratePreviewForURL` callback function implemented by a Quick Look generator may be called one or more times concurrently if the `QLSupportsConcurrentRequests` property in the generator bundle's `Info.plist` file is set to `true`. If this is the case, use the `preview` parameter to distinguish among current requests. The generator might also have the `QLNeedsToBeRunInMainThread` property set to `true`, in which case the callback is always invoked in the main thread.

Special Considerations

Thread-safety: For a discussions of issues and possible approaches, see “Overview of Generator Implementation” in *Quick Look Programming Guide*.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

`QuickLook/QLGenerator.h`

Data Types

QLPreviewRequestRef

An opaque reference representing an `QLPreviewRequest` object.

```
typedef struct __QLPreviewRequest *QLPreviewRequestRef;
```

Availability

Available in Mac OS X version 10.5 and later.

Declared In

`QLGenerator.h`

Constants

Preview Properties

Keys of the properties dictionary in functions used to create a preview.

```

QL_EXPORT const CFStringRef kQLPreviewPropertyDisplayNameKey;
QL_EXPORT const CFStringRef kQLPreviewPropertyWidthKey;
QL_EXPORT const CFStringRef kQLPreviewPropertyHeightKey;
QL_EXPORT const CFStringRef kQLPreviewPropertyStringEncodingKey;
QL_EXPORT const CFStringRef kQLPreviewPropertyMIMETYPEKey;
QL_EXPORT const CFStringRef kQLPreviewPropertyTextEncodingNameKey;
QL_EXPORT const CFStringRef kQLPreviewPropertyAttachmentDataKey;
QL_EXPORT const CFStringRef kQLPreviewPropertyAttachmentsKey;
QL_EXPORT const CFStringRef kQLPreviewContentIDScheme;

```

Constants

`kQLPreviewPropertyDisplayNameKey`

Specifies a custom display name in the preview panel. The default display name is the document title. The value must be encapsulated in a `CFString` object.

Available in Mac OS X v10.5 and later.

Declared in `QLGenerator.h`.

`kQLPreviewPropertyWidthKey`

Specifies the width (in points) of the preview. Note that this property is a hint; Quick Look might set the width automatically for some types of previews. The value must be encapsulated in a `CFNumber` object.

Available in Mac OS X v10.5 and later.

Declared in `QLGenerator.h`.

`kQLPreviewPropertyHeightKey`

Specifies the height (in points) of the preview. Note that this property is a hint; Quick Look might set the height automatically for some types of previews. The value must be encapsulated in a `CFNumber` object.

Available in Mac OS X v10.5 and later.

Declared in `QLGenerator.h`.

`kQLPreviewPropertyStringEncodingKey`

Specifies the string encoding (as an `CFStringEncoding` value) of the preview data if the native type is plain text. The value must be encapsulated in a `CFNumber` object.

Available in Mac OS X v10.5 and later.

Declared in `QLGenerator.h`.

`kQLPreviewPropertyMIMETYPEKey`

Gives the web content or attachment mime type. For the main data, the default type is “text/html”. The value is a `CFString` object.

Available in Mac OS X v10.5 and later.

Declared in `QLGenerator.h`.

`kQLPreviewPropertyTextEncodingNameKey`

Specifies the encoding of the web content or attachment text. For the value, use IANA encodings like “UTF-8”. The value is a `CFString` object.

Available in Mac OS X v10.5 and later.

Declared in `QLGenerator.h`.

`kQLPreviewPropertyAttachmentDataKey`

Gives the attachment data. The value is a `CFData` object.

Available in Mac OS X v10.5 and later.

Declared in `QLGenerator.h`.

`kQLPreviewPropertyAttachmentsKey`

Gives the list of attachments (or sub-resources). Value is a `CFDictionary` object.

The keys of the dictionary are the attachment identifiers (`CFString` objects) that can be referenced with the `cid:id` URL; the values are dictionaries using the `kQLPreviewPropertyAttachmentDataKey`, `kQLPreviewPropertyMIMETYPEKey` and `kQLPreviewPropertyTextEncodingNameKey` properties.

Available in Mac OS X v10.5 and later.

Declared in `QLGenerator.h`.

`kQLPreviewContentIDScheme`

The `cid` URL scheme.

The `cid` URL scheme permits the HTML of a mail message to refer to the images or other data included in the message. For more information go to <http://www.rfc-archive.org/getrfc.php?rfc=2111>.

Available in Mac OS X v10.5 and later.

Declared in `QLGenerator.h`.

Discussion

You use these keys to specify properties of the generated preview in the dictionary passed into the functions `QLPreviewRequestCreateContext` (page 12), `QLPreviewRequestCreatePDFContext` (page 13), and `QLPreviewRequestSetDataRepresentation` (page 16) as the final parameter.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

`QuickLook/QLGenerator.h`

QLThumbnailRequest Reference

Derived From:	CType
Framework:	QuickLook/QuickLook.h
Declared in	QLGenerator.h
Companion guide	Quick Look Programming Guide

Overview

Quick Look generators use the `QLThumbnailRequest` API to help create a thumbnail representation of a document that is returned to Quick Look clients (such as Finder and Spotlight) for display. An object of the `QLThumbnailRequest` opaque type represents a request from a client for a document thumbnail and is used to contain or locate the image the generator creates.

A generator must implement the [GenerateThumbnailForURL](#) (page 33) callback function to create and return a thumbnail image requested for a given document. `QLThumbnailRequest` gives you four ways to designate a thumbnail for a document:

- You can create a graphics context to draw the thumbnail in. Graphics contexts for thumbnails are of two kinds: bitmap and single-page vector. You use the [QLThumbnailRequestCreateContext](#) (page 27) for this purpose.
- You can extract the thumbnail from the document and return it using the [QLThumbnailRequestSetImageWithData](#) (page 31) function. For this approach, the document's application must save the thumbnail as part of the document data.
- You can create the thumbnail image and assign it to the thumbnail request using the [QLThumbnailRequestSetImage](#) (page 30) function.

Most of the other functions of `QLThumbnailRequest` let you get the data associated with the thumbnail request, such as the URL locating the document, the UTI identifying the content type of the document, and the maximum size allowed for the thumbnail image.

Note: This reference documentation also describes the use of the [QLThumbnailImageCreate](#) (page 25) function, which is called by *clients* of Quick Look, not by generator plug-ins. For example, an application could call this function to obtain a thumbnail image of a document that it is listing.

Functions by Task

Assigning content to the thumbnail request

[QLThumbnailRequestCreateContext](#) (page 27)

Creates a graphics context to draw the thumbnail in.

[QLThumbnailRequestSetImage](#) (page 30)

Sets the thumbnail request to a specified image.

[QLThumbnailRequestSetImageWithData](#) (page 31)

Sets the response to the thumbnail request to image data saved within the document.

[QLThumbnailRequestFlushContext](#) (page 28)

Flush the graphics context and sets the thumbnail response.

Getting attributes of the thumbnail request

[QLThumbnailRequestCopyContentUTI](#) (page 25)

Returns the UTI for the thumbnail request.

[QLThumbnailRequestCopyOptions](#) (page 26)

Returns the options specified for the thumbnail request.

[QLThumbnailRequestCopyURL](#) (page 27)

Returns the URL of the document for which the thumbnail request is requested.

[QLThumbnailRequestGetGeneratorBundle](#) (page 29)

Get the bundle of the generator receiving the thumbnail request.

[QLThumbnailRequestGetMaximumSize](#) (page 29)

Returns the maximum size (in points) specified for the thumbnail image.

[QLThumbnailRequestIsCancelled](#) (page 30)

Returns whether the thumbnail request has been cancelled by the client.

Getting the QLThumbnailRequest Type ID

[QLThumbnailRequestGetTypeID](#) (page 30)

Gets the type identifier for the `QLThumbnailRequest` opaque type.

Client Functions

The following functions are for use by clients of Quick Look (generally applications), not by generator plug-ins.

[QLThumbnailImageCreate](#) (page 25)

Creates a thumbnail image for the specified file.

Functions

QLThumbnailImageCreate

Creates a thumbnail image for the specified file.

```
QL_EXPORT CGImageRef QLThumbnailImageCreate(CFAllocatorRef allocator, CFURLRef url,
      CGSize maxThumbnailSize, CFDictionaryRef options);
```

Parameters

allocator

The allocator to use to create the thumbnail image.

url

The URL locating the file to create a thumbnail image for.

maxThumbnailSize

The maximum desired size of the thumbnail image.

options

A dictionary of options that affect the creation of the thumbnail image. Two options are supported: [kQLThumbnailOptionIconModeKey](#) (page 35) and [kQLThumbnailOptionScaleFactorKey](#) (page 34).

Return Value

Returns the thumbnail image or NULL if Quick Look does not support this file type

Discussion

This function does not supplant the use of Icon Services by applications as a way to get generic file icons and custom icons stored in the metadata fork of files.

Special Considerations

Thread-safety: This function is thread-safe so you can call it from any thread. However, because it is synchronous, you should generally call it in a background thread.

Availability

Available in Mac OS X v10.5 and later.

Declared In

QLThumbnailImage.h

QLThumbnailRequestCopyContentUTI

Returns the UTI for the thumbnail request.

```
QL_EXPORT CFStringRef QLThumbnailRequestCopyContentUTI(
    QLThumbnailRequestRef thumbnail
);
```

Parameters*thumbnail*

The thumbnail request object.

Return Value

The UTI identifying the content type of the document for which the thumbnail is requested; returns `NULL` if the UTI cannot be located. You should explicitly release this string object when it is no longer needed.

Special Considerations

Thread-safety: This function should be called in the same thread as the thumbnail request is made in; generally, this is the same thread in which the [GenerateThumbnailForURL](#) (page 33) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLThumbnailRequestCopyOptions

Returns the options specified for the thumbnail request.

```
QL_EXPORT CFDictionaryRef QLThumbnailRequestCopyOptions(
    QLThumbnailRequestRef thumbnail
);
```

Parameters*thumbnail*

A thumbnail request object.

Return Value

A dictionary containing the options that the client specified for the thumbnail request. See [“General Thumbnail Options”](#) (page 34) for supported options. You should explicitly release the dictionary when it is no longer needed.

Discussion

The client sets options in the thumbnail request to give hints to the generator. (For Mac OS X v10.5 no options are supported.) You should explicitly release the dictionary when it is no longer needed.

Special Considerations

Thread-safety: This function should be called in the same thread as the thumbnail request is made in; generally, this is the same thread in which the [GenerateThumbnailForURL](#) (page 33) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLThumbnailRequestCopyURL

Returns the URL of the document for which the thumbnail request is requested.

```

QL_EXPORT CFURLRef QLThumbnailRequestCopyURL(
    QLThumbnailRequestRef thumbnail
);

```

Parameters

thumbnail

The thumbnail request object.

Return Value

The URL identifying the document for which the thumbnail is requested. You should explicitly release the CFURL object when it is no longer needed.

Special Considerations

Thread-safety: This function should be called in the same thread as the thumbnail request is made in; generally, this is the same thread in which the [GenerateThumbnailForURL](#) (page 33) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLThumbnailRequestCreateContext

Creates a graphics context to draw the thumbnail in.

```

QL_EXPORT CGContextRef QLThumbnailRequestCreateContext(
    QLThumbnailRequestRef preview,
    CGSize size,
    Boolean isBitmap,
    CFDictionaryRef properties
);

```

Parameters

preview

The thumbnail request object.

size

The size of the thumbnail; if *isBitmap* is *true* the size is in pixels, otherwise it is in points.

isBitmap

true if the thumbnail data is bitmap-based, *false* if vector-based. This value of this parameter affects the interpretation of the *size* parameter.

properties

A dictionary containing properties for the thumbnail response. (For Mac OS X v10.5 no properties have been defined.).

Return Value

A Core Graphics graphics-context object that you can draw your thumbnail image in. You should explicitly release this object when it is no longer needed.

Discussion

You can directly draw your thumbnail data in the graphics-context object created by this function. After calling this function, you should flush the context with [QLThumbnailRequestFlushContext](#) (page 28). Also be sure to release the `CGContext` object.

With this function you can create two types of graphics contexts for drawing thumbnails: bitmap and single-page vector-based; you use the `isBitmap` flag to distinguish between the two. Quick Look handles bitmap thumbnail context differently than non-bitmap contexts; in the latter case, Quick Look might scale the drawing up or down if necessary, and it respects the scale factor (for HiDPI support).

If you prefer to work in Objective-C code, you can convert the created `CGContextRef` to a `NSGraphicsContext` object using `graphicsContextWithGraphicsPort:flipped:`.

Special Considerations

Thread-safety: This function should be called in the same thread as the thumbnail request is made in; generally, this is the same thread in which the [GenerateThumbnailForURL](#) (page 33) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Related Sample Code

QuickLookSketch

Declared In

QLGenerator.h

QLThumbnailRequestFlushContext

Flush the graphics context and sets the thumbnail response.

```
QL_EXPORT void QLThumbnailRequestFlushContext(
    QLThumbnailRequestRef preview,
    CGContextRef context
);
```

Parameters

preview

The preview request object.

context

The graphics context to flush.

Discussion

You should call this method after drawing in the graphics context created by [QLThumbnailRequestCreateContext](#) (page 27).

Special Considerations

Thread-safety: This function should be called in the same thread as the thumbnail request is made in; generally, this is the same thread in which the [GenerateThumbnailForURL](#) (page 33) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Related Sample Code

QuickLookSketch

Declared In

QLGenerator.h

QLThumbnailRequestGetGeneratorBundle

Get the bundle of the generator receiving the thumbnail request.

```
QL_EXPORT CFBundleRef QLThumbnailRequestGetGeneratorBundle(  
    QLThumbnailRequestRef thumbnail  
);
```

Parameters*thumbnail*

The thumbnail request object.

Return Value

A reference to the bundle object representing the generator's bundle.

Special Considerations

Thread-safety: This function should be called in the same thread as the thumbnail request is made in; generally, this is the same thread in which the [GenerateThumbnailForURL](#) (page 33) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLThumbnailRequestGetMaximumSize

Returns the maximum size (in points) specified for the thumbnail image.

```
QL_EXPORT CGSize QLThumbnailRequestGetMaximumSize(  
    QLThumbnailRequestRef thumbnail  
);
```

Parameters*thumbnail*

The thumbnail request object.

Return Value

The maximum size (in points) specified for the thumbnail image.

Special Considerations

Thread-safety: This function should be called in the same thread as the thumbnail request is made in; generally, this is the same thread in which the [GenerateThumbnailForURL](#) (page 33) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLThumbnailRequestGetTypeID

Gets the type identifier for the `QLThumbnailRequest` opaque type.

```
QL_EXPORT CTypeID QLThumbnailRequestGetTypeID();
```

Return Value

The type identifier for the `QLThumbnailRequest` opaque type.

Special Considerations

Thread-safety: This function should be called in the same thread as the thumbnail request is made in; generally, this is the same thread in which the [GenerateThumbnailForURL](#) (page 33) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

`QLGenerator.h`

QLThumbnailRequestIsCancelled

Returns whether the thumbnail request has been cancelled by the client.

```
QL_EXPORT Boolean QLThumbnailRequestIsCancelled(
    QLThumbnailRequestRef thumbnail
);
```

Parameters

thumbnail

The thumbnail request object.

Return Value

`true` if the request is being cancelled, `false` otherwise.

Special Considerations

Thread-safety: This function should be called in the same thread as the thumbnail request is made in; generally, this is the same thread in which the [GenerateThumbnailForURL](#) (page 33) callback was invoked.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`QLGenerator.h`

QLThumbnailRequestSetImage

Sets the thumbnail request to a specified image.

```

QL_EXPORT void QLThumbnailRequestSetImage(
    QLThumbnailRequestRef thumbnail,
    CGImageRef image,
    CFDictionaryRef properties);
);

```

Parameters*thumbnail*

The thumbnail request object.

image

The image object to be used as the thumbnail of the document.

properties

A dictionary of properties for the thumbnail. (For Mac OS X v10.5 no properties have been defined.)

Discussion

You call this function to have Quick Look use the `CGImage` object supplied in *image* as the thumbnail. Call [QLThumbnailRequestGetMaximumSize](#) (page 29) to get the maximum allowed size for thumbnail and resize it if necessary before calling `QLThumbnailSetImage`.

Special Considerations

Thread-safety: This function should be called in the same thread as the thumbnail request is made in; generally, this is the same thread in which the [GenerateThumbnailForURL](#) (page 33) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

QLGenerator.h

QLThumbnailRequestSetImageWithData

Sets the response to the thumbnail request to image data saved within the document.

```

QL_EXPORT void QLThumbnailRequestSetImageWithData(
    QLThumbnailRequestRef thumbnail,
    CFDataRef data,
    CFDictionaryRef properties);
);

```

Parameters*thumbnail*

The thumbnail request object.

data

The image data, which must be in a format supported by the Image I/O framework (JPG, PNG, and so on). In other words, a content type of `kUTTypeImage` is assumed. (`ImageIO.framework` is a subframework of the umbrella Application Services framework.)

properties

A dictionary of properties. The only property that you can currently specify is `kCGImageSourceTypeIdentifierHint`; see *Quartz 2D Programming Guide* for information about this property.

Discussion

This function returns the thumbnail as a `CFData` object containing image data. The document's application must save this data as part of the document's data; the generator retrieves it and uses this function to return it to the client. Before you call this function, call [QLThumbnailRequestGetMaximumSize](#) (page 29) to obtain the maximum allowed size for the thumbnail and resize the image if necessary.

Special Considerations

Thread-safety: This function should be called in the same thread as the thumbnail request is made in; generally, this is the same thread in which the [GenerateThumbnailForURL](#) (page 33) callback was invoked.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

`QLGenerator.h`

Callbacks

CancelThumbnailGeneration

Defines a pointer to a function that Quick Look calls to cancel a thumbnail request.

```
void (*CancelThumbnailGeneration)(
    void *thisInterface,
    QLThumbnailThumbnailRef thumbnail
);
```

Parameters

thisInterface

The `CFPlugIn` COM-style interface for the generator.

thumbnail

The object representing the thumbnail request.

Discussion

If the client application decides (usually because of user choice) that it is no longer interested in the thumbnail currently being created by a Quick Look generator, it invokes this callback function. The generator can implement this function to stop creating the thumbnail and clean up the resources used for creating it.

An alternative to implementing this callback is to periodically poll the [QLThumbnailRequestRef](#) (page 34) object with [QLThumbnailRequestIsCancelled](#) (page 30) to see if the request has been canceled.

Important: Because this function is called on a thread different from the one used to request that the thumbnail be created, you should be extra careful about thread safety. If you have any doubts about thread safety, do not implement this callback.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

`QuickLook/QLGenerator.h`

GenerateThumbnailForURL

Defines a pointer to the callback function that Quick Look calls to request a thumbnail from a generator.

```
OSStatus (*GenerateThumbnailForURL)(
    void *thisInterface,
    QLThumbnailRequestRef thumbnail,
    CFURLRef url,
    CFStringRef contentTypeUTI,
    CFDictionaryRef options,
    CGSize maxSize
);
```

The Xcode template for Quick Look generators automatically creates a “skeletal” function with the same name as the callback symbol name: `GenerateThumbnailForURL`.

Parameters

thisInterface

The `CFPlugIn` COM-style interface for the generator.

thumbnail

The object containing all information relevant to the thumbnail request. The generator’s role is to assign a document thumbnail (in one of the native Quick Look types) to this object before it returns.

url

A URL (represented by a `CFURLRef` object) that locates the document for which a thumbnail is requested.

contentTypeUTI

The UTI specifying the content type of the document for which the thumbnail is requested.

options

A dictionary of options passed from the client (for example, Finder or Spotlight) that are hints for processing the thumbnail.

maxSize

The maximum desired size of the thumbnail.

Return Value

A status code representing the result of the request. For the current version of this callback, you should always return `noErr`.

Discussion

The `GenerateThumbnailForURL` callback function implemented by a Quick Look generator may be called one or more times concurrently if the `QLSupportsConcurrentRequests` option in the generator bundle’s `Info.plist` file is set to `true`. If this is the case, use the *thumbnail* parameter to distinguish among current requests. The generator might also have the `QLNeedsToBeRunInMainThread` property set to `true`, in which case the callback is always invoked in the main thread.

Special Considerations

Thread-safety: For a discussions of issues and possible approaches, see “Overview of Generator Implementation” in *Quick Look Programming Guide*.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

`QuickLook/QLGenerator.h`

Data Types

QLThumbnailRequestRef

An opaque reference representing an `QLThumbnailRequest` object.

```
typedef struct __QLThumbnailRequest *QLThumbnailRequestRef;
```

Availability

Available in Mac OS X version 10.5 and later.

Declared In

`QLGenerator.h`

Constants

General Thumbnail Options

Keys used for the options passed by the client to the generator and by the client when requesting a thumbnail image.

```
QL_EXPORT const CFStringRef kQLThumbnailOptionScaleFactorKey;
```

Constants

`kQLThumbnailOptionScaleFactorKey`

A key used to specify the user scale factor for the thumbnail. The `float` value is encapsulated in a `CFNumber` object. If this option is absent, the default value is 1.0

Available in Mac OS X v10.5 and later.

Declared in `QLThumbnailImage.h`.

Discussion

This constant is a key for accessing the scale-factor value in the dictionary passed into [GenerateThumbnailForURL](#) (page 33) in the `options` parameter. It can also be used for the [QLThumbnailImageCreate](#) (page 25) function, which is called by clients (generally applications) of Quick Look.

Availability

Available in Mac OS X version 10.5 and later.

Declared In

`QuickLook/QLThumbnailImage.h`

Client-Specific Thumbnail Options

Clients of Quick Look can specify the following option-dictionary properties for special handling of thumbnail images.

```
QL_EXPORT const CFStringRef kQLThumbnailOptionIconModeKey;
```

Constants

`kQLThumbnailOptionIconModeKey`

If the value is `kCFBooleanTrue`, Quick Look produces the thumbnail as an icon. The image includes all the typical icon decor, such as shadows and a curled corner. If `kCFBooleanFalse` (the default) is specified, no icon decor is included with the thumbnail image.

Available in Mac OS X v10.5 and later.

Declared in `QLThumbnailImage.h`.

Declared In

`QuickLook/QLThumbnailImage.h`

Document Revision History

This table describes the changes to *Quick Look Framework Reference*.

Date	Notes
2007-04-13	New document that describes the Quick Look framework functions, types, and constants used for creating Quick Look generator plug-ins.

REVISION HISTORY

Document Revision History

Index

C

CancelPreviewGeneration [callback](#) 17
CancelThumbnailGeneration [callback](#) 32
Client-Specific Thumbnail Options 34

G

General Thumbnail Options 34
GeneratePreviewForURL [callback](#) 18
GenerateThumbnailForURL [callback](#) 33

K

kQLPreviewContentIDScheme [constant](#) 21
kQLPreviewPropertyAttachmentDataKey [constant](#) 20
kQLPreviewPropertyAttachmentsKey [constant](#) 21
kQLPreviewPropertyDisplayNameKey [constant](#) 20
kQLPreviewPropertyHeightKey [constant](#) 20
kQLPreviewPropertyMIMETYPEKey [constant](#) 20
kQLPreviewPropertyStringEncodingKey [constant](#) 20
kQLPreviewPropertyTextEncodingNameKey [constant](#) 20
kQLPreviewPropertyWidthKey [constant](#) 20
kQLThumbnailOptionIconModeKey [constant](#) 35
kQLThumbnailOptionScaleFactorKey [constant](#) 34

P

Preview Properties 19

Q

QLPreviewRequestCopyContentUTI [function](#) 10
QLPreviewRequestCopyOptions [function](#) 11
QLPreviewRequestCopyURL [function](#) 12
QLPreviewRequestCreateContext [function](#) 12
QLPreviewRequestCreatePDFContext [function](#) 13
QLPreviewRequestFlushContext [function](#) 14
QLPreviewRequestGetGeneratorBundle [function](#) 15
QLPreviewRequestGetTypeID [function](#) 15
QLPreviewRequestIsCancelled [function](#) 15
QLPreviewRequestRef [structure](#) 19
QLPreviewRequestSetDataRepresentation [function](#) 16
QLThumbnailImageCreate [function](#) 25
QLThumbnailRequestCopyContentUTI [function](#) 25
QLThumbnailRequestCopyOptions [function](#) 26
QLThumbnailRequestCopyURL [function](#) 27
QLThumbnailRequestCreateContext [function](#) 27
QLThumbnailRequestFlushContext [function](#) 28
QLThumbnailRequestGetGeneratorBundle [function](#) 29
QLThumbnailRequestGetMaximumSize [function](#) 29
QLThumbnailRequestGetTypeID [function](#) 30
QLThumbnailRequestIsCancelled [function](#) 30
QLThumbnailRequestRef [structure](#) 34
QLThumbnailRequestSetImage [function](#) 30
QLThumbnailRequestSetImageWithData [function](#) 31