
NSURL Class Reference

Data Management



2009-10-09



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, iChat, Mac, Mac OS, and Quartz are trademarks of Apple Inc., registered in the United States and other countries.

Finder is a trademark of Apple Inc.

Java and all Java-based trademarks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

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

NSURL Class Reference 7

Overview	7
Adopted Protocols	8
Tasks	8
Creating an NSURL	8
Identifying and Comparing Objects	9
Querying an NSURL	9
Loading the Resource of an NSURL Object	9
Accessing the Parts of the URL	10
Modifying and Converting a File URL	11
Working with Bookmark Data	11
Getting and Setting File System Resource Properties	11
Class Methods	12
bookmarkDataWithContentsOfURL:error:	12
fileURLWithPath:	12
fileURLWithPath:isDirectory:	13
fileURLWithPathComponents:	14
resourceValuesForKeys:fromBookmarkData:	14
URLByResolvingBookmarkData:options:relativeToURL:bookmarkDataIsStale:error:	15
URLWithString:	15
URLWithString:relativeToURL:	16
writeBookmarkData:toURL:options:error:	16
Instance Methods	17
absoluteString	17
absoluteURL	18
baseURL	18
bookmarkDataWithOptions:includingResourceValuesForKeys:relativeToURL:error:	18
checkResourcesReachableAndReturnError:	19
filePathURL	19
fileReferenceURL	19
fragment	20
getResourceValue:forKey:error:	20
host	21
initWithResolvingBookmarkData:options:relativeToURL:bookmarkDataIsStale:error:	21
initWithFileURLWithPath:	22
initWithFileURLWithPath:isDirectory:	22
initWithScheme:host:path:	23
initWithString:	23
initWithString:relativeToURL:	24
isEqual:	25
isFileReferenceURL	25

isFileURL	25
lastPathComponent	25
parameterString	26
password	26
path	26
pathComponents	27
pathExtension	27
port	27
query	28
relativePath	28
relativeString	28
resourceSpecifier	29
resourceValuesForKeys:error:	29
scheme	29
setResourceValue:forKey:error:	30
setResourceValues:error:	30
standardizedURL	31
URLByAppendingPathComponent:	31
URLByAppendingPathExtension:	32
URLByDeletingLastPathComponent	32
URLByDeletingPathExtension	32
URLByResolvingSymlinksInPath	33
URLByStandardizingPath	33
user	33
Constants	34
NSURL Schemes	34
Common File System Resource Keys	34
File Property Keys	38
Volume Property Keys	38
Bookmark Data Creation Options	40
Bookmark Data Resolution Options	41
NSURLHandle FTP Property Keys	41
NSURLHandle HTTP Property Keys	43

Appendix A **Deprecated NSURL Methods** 45

Deprecated in Mac OS X v10.4	45
loadResourceDataNotifyingClient:usingCache:	45
propertyForKey:	45
resourceDataUsingCache:	46
setProperty:forKey:	46
setResourceData:	47
URLHandleUsingCache:	47

Document Revision History 49

Index 51

NSURL Class Reference

Inherits from	NSObject
Conforms to	NSCoding NSCopying NSURLHandleClient NSObject (NSObject)
Framework	/System/Library/Frameworks/Foundation.framework
Availability	Available in Mac OS X v10.0 and later.
Companion guide	URL Loading System
Declared in	NSURL.h NSURLHandle.h
Related sample code	CoreRecipes iChatStatusFromApplication ImageClient ImageKitDemo LSMSmartCategorizer

Overview

The NSURL class provides a way to manipulate URLs and the resources they reference. NSURL objects understand URLs as specified in RFCs 1808, 1738, and 2732. The litmus test for conformance to RFC 1808 is as recommended in RFC 1808—whether the first two characters of `resourceSpecifier` (page 29) are `@"/"`.

NSURL objects can be used to refer to files, and are the preferred way to do so. ApplicationKit objects that can read data from or write data to a file generally have methods that accept an NSURL object instead of a pathname as the file reference. NSWorkspace provides `openURL:` to open a location specified by a URL. To get the contents of a URL, NSString provides `stringWithContentsOfURL:` and NSData provides `dataWithContentsOfURL:`.

An NSURL object is composed of two parts—a potentially `nil` base URL and a string that is resolved relative to the base URL. An NSURL object whose string is fully resolved without a base is considered absolute; all others are considered relative.

The NSURL class will fail to create a new NSURL object if the path being passed is not well-formed—the path must comply with RFC 2396. Examples of cases that will not succeed are strings containing space characters and high-bit characters. Should creating an NSURL object fail, the creation methods return `nil`, which you must be prepared to handle. If you are creating NSURL objects using file system paths, you should use

[fileURLWithPath:](#) (page 12) or [initWithFileURLWithPath:](#) (page 22), which handle the subtle differences between URL paths and file system paths. If you wish to be tolerant of malformed path strings, you'll need to use functions provided by the Core Foundation framework to clean up the strings.

The classes `NSURLConnection` and `NSURLSessionDownload` define methods useful for loading URL resources in the background. See *URL Loading System* for more information

See also NSURL Additions Reference in the Application Kit framework, which add methods supporting pasteboards.

NSURL is “toll-free bridged” with its Core Foundation counterpart, *CFURL Reference*. This means that the Core Foundation type is interchangeable in function or method calls with the bridged Foundation object, providing you cast one type to the other. In an API where you see an `NSURL *` parameter, you can pass in a `CFURLRef`, and in an API where you see a `CFURLRef` parameter, you can pass in a pointer to an `NSURL` instance. This approach also applies to your concrete subclasses of `NSURL`. See *Interchangeable Data Types* for more information on toll-free bridging.

Adopted Protocols

NSCoding

- `encodeWithCoder:`
- `initWithCoder:`

NSCopying

- `copyWithZone:`

NSURLSessionHandleClient

- `URLHandleResourceDidBeginLoading:`
- `URLHandleResourceDidCancelLoading:`
- `URLHandleResourceDidFinishLoading:`
- `URLHandle:resourceDataDidBecomeAvailable:`
- `URLHandle:resourceDidFailLoadingWithReason:`

Tasks

Creating an NSURL

- [initWithScheme:host:path:](#) (page 23)
Initializes a newly created `NSURL` with a specified scheme, host, and path.
- + [URLWithString:](#) (page 15)
Creates and returns an `NSURL` object initialized with a provided string.
- [initWithString:](#) (page 23)
Initializes an `NSURL` object with a provided string.
- + [URLWithString:relativeToURL:](#) (page 16)
Creates and returns an `NSURL` object initialized with a base URL and a relative string.

- [initWithString:relativeToURL:](#) (page 24)
Initializes an NSURL object with a base URL and a relative string.
- + [fileURLWithPath:isDirectory:](#) (page 13)
Initializes and returns a newly created NSURL object as a file URL with a specified path.
- [initWithFileURLWithPath:isDirectory:](#) (page 22)
Initializes a newly created NSURL referencing the local file or directory at *path*.
- + [fileURLWithPath:](#) (page 12)
Initializes and returns a newly created NSURL object as a file URL with a specified path.
- [initWithFileURLWithPath:](#) (page 22)
Initializes a newly created NSURL referencing the local file or directory at *path*.
- + [fileURLWithPathComponents:](#) (page 14)
Initializes and returns a newly created NSURL object as a file URL with specified path components.
- + [URLByResolvingBookmarkData:options:relativeToURL:bookmarkDataIsStale:error:](#) (page 15)
Returns a new URL made by resolving bookmark data.
- [initWithResolvingBookmarkData:options:relativeToURL:bookmarkDataIsStale:error:](#) (page 21)
Initializes a newly created NSURL that points to a location specified by resolving bookmark data.

Identifying and Comparing Objects

- [isEqual:](#) (page 25)
Returns a Boolean value that indicates whether the receiver and a given object are equal.

Querying an NSURL

- [checkResourceIsReachableAndReturnError:](#) (page 19)
Returns whether the resource pointed to by a file URL can be reached.
- [isFileReferenceURL:](#) (page 25)
Returns whether the URL is a file reference URL.
- [isFileURL:](#) (page 25)
Returns whether the receiver uses the file scheme.

Loading the Resource of an NSURL Object

- [loadResourceDataNotifyingClient:usingCache:](#) (page 45) **Deprecated in Mac OS X v10.4**
Loads the receiver's resource data in the background.
- [propertyForKey:](#) (page 45) **Deprecated in Mac OS X v10.4**
Returns the specified property of the receiver's resource.
- [resourceDataUsingCache:](#) (page 46) **Deprecated in Mac OS X v10.4**
Returns the receiver's resource data, loading it if necessary. Use `NSURLConnection` instead of this method.

- `setProperty:forKey:` (page 46) **Deprecated in Mac OS X v10.4**
Changes the specified property of the receiver's resource.
- `setResourceData:` (page 47) **Deprecated in Mac OS X v10.4**
Attempts to set the resource data for the receiver.
- `URLHandleUsingCache:` (page 47) **Deprecated in Mac OS X v10.4**
Returns a URL handle to service the receiver.

Accessing the Parts of the URL

- `absoluteString` (page 17)
Returns the string for the receiver as if it were an absolute URL.
- `absoluteURL` (page 18)
Returns an absolute URL that refers to the same resource as the receiver.
- `baseURL` (page 18)
Returns the base URL of the receiver.
- `fragment` (page 20)
Returns the fragment of a URL conforming to RFC 1808.
- `host` (page 21)
Returns the host of a URL conforming to RFC 1808.
- `lastPathComponent` (page 25)
Returns the last path component of a file URL.
- `parameterString` (page 26)
Returns the parameter string of a URL conforming to RFC 1808.
- `password` (page 26)
Returns the password of a URL conforming to RFC 1808.
- `path` (page 26)
Returns the path of a URL conforming to RFC 1808.
- `pathComponents` (page 27)
Returns the individual path components of a file URL in an array.
- `pathExtension` (page 27)
Returns the path extension of a file URL.
- `port` (page 27)
Returns the port number of a URL conforming to RFC 1808.
- `query` (page 28)
Returns the query of a URL conforming to RFC 1808.
- `relativePath` (page 28)
Returns the path of a URL conforming to RFC 1808, without resolving against the receiver's base URL.
- `relativeString` (page 28)
Returns a string representation of the relative portion of the URL.
- `resourceSpecifier` (page 29)
Returns the resource specifier of the URL.
- `scheme` (page 29)
Returns the scheme of the URL.

- [standardizedURL](#) (page 31)
Returns a new NSURL object with any instances of "." or "." removed from its path.
- [user](#) (page 33)
Returns the user portion of a URL conforming to RFC 1808.

Modifying and Converting a File URL

- [filePathURL](#) (page 19)
Returns a new file path URL that points to the same resource as the original URL.
- [fileReferenceURL](#) (page 19)
Returns a new file reference URL that points to the same resource as the original URL.
- [URLByAppendingPathComponent:](#) (page 31)
Returns a new URL made by appending a path component to the original URL.
- [URLByAppendingPathExtension:](#) (page 32)
Returns a new URL made by appending a path extension to the original URL.
- [URLByDeletingLastPathComponent](#) (page 32)
Returns a new URL made by deleting the last path component from the original URL.
- [URLByDeletingPathExtension](#) (page 32)
Returns a new URL made by deleting the path extension, if any, from the original URL.
- [URLByResolvingSymlinksInPath](#) (page 33)
Returns a new URL that points to the same resource as the original URL and includes no symbolic links.
- [URLByStandardizingPath](#) (page 33)
Returns a new URL that points to the same resource as the original URL and is an absolute path.

Working with Bookmark Data

- + [bookmarkDataWithContentsOfURL:error:](#) (page 12)
Initializes and returns bookmark data derived from an alias file pointed to by a specified URL.
- [bookmarkDataWithOptions:includingResourceValuesForKeys:relativeToURL:error:](#) (page 18)
Returns bookmark data for the URL, created with specified options and resource values.
- + [writeBookmarkData:toURL:options:error:](#) (page 16)
Creates an alias file on disk at a specified location with specified bookmark data.

Getting and Setting File System Resource Properties

- [getResourceValue:forKey:error:](#) (page 20)
Returns the resource value for the property identified by a given key.
- [resourceValuesForKeys:error:](#) (page 29)
Returns the resource values for the properties identified by specified array of keys.
- [setResourceValue:forKey:error:](#) (page 30)
Sets the resource property of the URL specified by a given key to a given value.

- + [resourceValuesForKeys:fromBookmarkData:](#) (page 14)
Returns the resource values for properties identified by a specified array of keys contained in specified bookmark data.
- [setResourceValues:error:](#) (page 30)
Sets resource properties of the URL specified by a given set of keys to a given set of values.

Class Methods

bookmarkDataWithContentsOfURL:error:

Initializes and returns bookmark data derived from an alias file pointed to by a specified URL.

```
+ (NSData *)bookmarkDataWithContentsOfURL:(NSURL *)bookmarkFileURL error:(NSError **)error
```

Parameters

bookmarkFileURL

The URL that points to the alias file.

error

The error that occurred in the case that the bookmark data cannot be derived.

Return Value

The bookmark data for the alias file.

Discussion

If *bookmarkFileURL* points to an alias file created prior to Mac OS X v10.6 that contains Alias Manager information but no bookmark data, this method synthesizes bookmark data for the file.

This method returns `nil` if bookmark data cannot be created.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

fileURLWithPath:

Initializes and returns a newly created NSURL object as a file URL with a specified path.

```
+ (id)fileURLWithPath:(NSString *)path
```

Parameters

path

The path that the NSURL object will represent. *path* should be a valid system path. If *path* begins with a tilde, it must first be expanded with `stringByExpandingTildeInPath`.

Passing `nil` for this parameter produces an exception.

Return Value

An NSURL object initialized with *path*.

Discussion

This method assumes that *path* is a directory if it ends with a slash. If *path* does not end with a slash, the method examines the file system to determine if *path* is a file or a directory. If *path* exists in the file system and is a directory, the method appends a trailing slash. If *path* does not exist in the file system, the method assumes that it represents a file and does not append a trailing slash.

As an alternative, consider using [fileURLWithPath:isDirectory:](#) (page 13), which allows you to explicitly specify whether the returned NSURL object represents a file or directory.

Availability

Available in Mac OS X v10.0 and later.

See Also

[initWithFileURLWithPath:](#) (page 22)

Related Sample Code

CoreRecipes

FunHouse

ImageKitDemo

Quartz Composer WWDC 2005 TextEdit

SimpleStickies

Declared In

NSURL.h

fileURLWithPath:isDirectory:

Initializes and returns a newly created NSURL object as a file URL with a specified path.

```
+ (id)fileURLWithPath:(NSString *)path isDirectory:(BOOL)isDir
```

Parameters

path

The path that the NSURL object will represent. *path* should be a valid system path. If *path* begins with a tilde, it must first be expanded with `stringByExpandingTildeInPath`.

Passing `nil` for this parameter produces an exception.

isDir

A Boolean value that specifies whether *path* is treated as a directory path when resolving against relative path components. Pass `YES` if the *path* indicates a directory, `NO` otherwise.

Return Value

An NSURL object initialized with *path*.

Availability

Available in Mac OS X v10.5 and later.

See Also

[initWithFileURLWithPath:](#) (page 22)

Related Sample Code

AnimatedTableView

Denoise

From A View to A Movie
 IconCollection
 IKSideshowDemo

Declared In

NSURL.h

fileURLWithPathComponents:

Initializes and returns a newly created NSURL object as a file URL with specified path components.

```
+ (NSURL *)fileURLWithPathComponents:(NSArray *)components
```

Parameters

components

An array of path components.

Passing *nil* for this parameter produces an exception.

Return Value

An NSURL object initialized with *components*.

Discussion

The path components are separated by a forward slash in the returned URL.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

resourceValuesForKeys:fromBookmarkData:

Returns the resource values for properties identified by a specified array of keys contained in specified bookmark data.

```
+ (NSDictionary *)resourceValuesForKeys:(NSArray *)keys fromBookmarkData:(NSData *)bookmarkData
```

Parameters

keys

An array of names of URL resource properties.

bookmarkData

The bookmark data the resource values are derived from.

Return Value

A dictionary of the requested resource values contained in *bookmarkData*.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

URLByResolvingBookmarkData:options:relativeToURL:bookmarkDataIsStale:error:

Returns a new URL made by resolving bookmark data.

```
+ (id)URLByResolvingBookmarkData:(NSData *)bookmarkData
    options:(NSURLBookmarkResolutionOptions)options relativeToURL:(NSURL
    *)relativeURL bookmarkDataIsStale:(BOOL *)isStale error:(NSError **)error
```

Parameters

bookmarkData

The bookmark data the URL is derived from.

options

Options taken into account when resolving the bookmark data.

relativeURL

The base URL that the bookmark data is relative to.

isStale

If YES, the bookmark data is stale.

error

The error that occurred in the case that the URL cannot be created.

Return Value

A new URL made by resolving *bookmarkData*.

Availability

Available in Mac OS X v10.6 and later.

Related Sample Code

QuickLookDownloader

Declared In

NSURL.h

URLWithString:

Creates and returns an NSURL object initialized with a provided string.

```
+ (id)URLWithString:(NSString *)URLString
```

Parameters

URLString

The string with which to initialize the NSURL object. Must conform to RFC 2396. This method parses *URLString* according to RFCs 1738 and 1808.

Return Value

An NSURL object initialized with *URLString*. If the string was malformed, returns *nil*.

Discussion

This method expects *URLString* to contain any necessary percent escape codes, which are `‘:’, ‘/’, ‘%’, ‘#’, ‘;’, and ‘@’`. Note that `‘%’` escapes are translated via UTF-8.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

LSMSmartCategorizer
 NewsReader
 ObjectPath
 VertexPerformanceTest
 With and Without Bindings

Declared In

NSURL.h

URLWithString:relativeToURL:

Creates and returns an NSURL object initialized with a base URL and a relative string.

```
+ (id)URLWithString:(NSString *)URLString relativeToURL:(NSURL *)baseURL
```

Parameters

URLString

The string with which to initialize the NSURL object. May not be `nil`. Must conform to RFC 2396. *URLString* is interpreted relative to *baseURL*.

baseURL

The base URL for the NSURL object.

Return Value

An NSURL object initialized with *URLString* and *baseURL*. If *URLString* was malformed, returns `nil`.

Discussion

This method expects *URLString* to contain any necessary percent escape codes.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CocoaHTTPServer
 CocoaSlides
 CocoaSOAP
 Reducer

Declared In

NSURL.h

writeBookmarkData:toURL:options:error:

Creates an alias file on disk at a specified location with specified bookmark data.

```
+ (BOOL)writeBookmarkData:(NSData *)bookmarkData toURL:(NSURL *)bookmarkFileURL
options:(NSURLBookmarkFileCreationOptions)options error:(NSError **)error
```

Parameters

bookmarkData

The bookmark data containing information for the alias file.

bookmarkFileURL

The desired location of the alias file.

options

Options taken into account when creating the alias file.

error

The error that occurred in the case that the alias file cannot be created.

Return Value

YES if the alias file is successfully created; otherwise, NO.

Discussion

This method will produce an error if *bookmarkData* was not created with the `NSURLBookmarkCreationSuitableForBookmarkFile` option.

If *bookmarkFileURL* points to a directory, the alias file will be created in that directory with its name derived from the information in *bookmarkData*. If *bookmarkFileURL* points to a file, the alias file will be created with the location and name indicated by *bookmarkFileURL*, and its extension will be changed to `.alias` if it is not already.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

Instance Methods

absoluteString

Returns the string for the receiver as if it were an absolute URL.

```
- (NSString *)absoluteString
```

Return Value

An absolute string for the URL. Creating by resolving the receiver's string against its base according to the algorithm given in RFC 1808.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

NewsReader

PDFKitLinker2

SourceView

With and Without Bindings

XMLBrowser

Declared In

NSURL.h

absoluteURL

Returns an absolute URL that refers to the same resource as the receiver.

```
- (NSURL *)absoluteURL
```

Return Value

An absolute URL that refers to the same resource as the receiver. If the receiver is already absolute, returns `self`. Resolution is performed per RFC 1808.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSURL.h

baseURL

Returns the base URL of the receiver.

```
- (NSURL *)baseURL
```

Return Value

The base URL of the receiver. If the receiver is an absolute URL, returns `nil`.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

JavaFrameEmbedding example

Declared In

NSURL.h

bookmarkDataWithOptions:includingResourceValuesForKeys:relativeToURL:error:

Returns bookmark data for the URL, created with specified options and resource values.

```
- (NSData *)bookmarkDataWithOptions:(NSURLBookmarkCreationOptions)options
    includingResourceValuesForKeys:(NSArray *)keys relativeToURL:(NSURL *)relativeURL
    error:(NSError **)error
```

Parameters

options

Options taken into account when creating the bookmark data.

keys

An array of names of URL resource properties.

relativeURL

The URL that the bookmark data will be relative to.

error

The error that occurred in the case that the bookmark data cannot be created.

Return Value

The bookmark data for the URL.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

checkResourceIsReachableAndReturnError:

Returns whether the resource pointed to by a file URL can be reached.

```
- (BOOL)checkResourceIsReachableAndReturnError:(NSError **)error
```

Parameters

error

The error that occurred in the case that the resource cannot be reached.

Return Value

YES if the resource is reachable; otherwise, NO.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

filePathURL

Returns a new file path URL that points to the same resource as the original URL.

```
- (NSURL *)filePathURL
```

Return Value

The new file path URL.

Discussion

If the original URL is a file reference URL, this method converts it to a file path URL. If the original URL is a file path URL, the returned URL is identical. If the original URL is not a file URL, this method returns `nil`.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

fileReferenceURL

Returns a new file reference URL that points to the same resource as the original URL.

```
- (NSURL *)fileReferenceURL
```

Return Value

The new file reference URL.

Discussion

If the original URL is a file path URL, this method converts it to a file reference URL. If the original URL is a file reference URL, the returned URL is identical. If the original URL is not a file URL, this method returns `nil`.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

fragment

Returns the fragment of a URL conforming to RFC 1808.

```
- (NSString *)fragment
```

Return Value

The fragment of the URL. If the receiver does not conform to RFC 1808, returns `nil`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSURL.h

getResourceValue:forKey:error:

Returns the resource value for the property identified by a given key.

```
- (BOOL)getResourceValue:(id *)value forKey:(NSString *)key error:(NSError **)error
```

Parameters

value

The value for the property identified by *key*.

key

The name of one of the URL's resource properties.

error

The error that occurred in the case that the resource value cannot be retrieved.

Return Value

YES if *value* is successfully populated; otherwise, NO.

Discussion

value is set to `nil` if the requested resource value is not defined for the URL.

Availability

Available in Mac OS X v10.6 and later.

Related Sample Code

TextSizingExample

Declared In

NSURL.h

host

Returns the host of a URL conforming to RFC 1808.

- (NSString *)host

Return ValueThe host of the URL. If the receiver does not conform to RFC 1808, returns `nil`.**Availability**

Available in Mac OS X v10.0 and later.

Related Sample Code

CoreRecipes

Declared In

NSURL.h

initWithResolvingBookmarkData:options:relativeToURL:bookmarkDataIsStale:error:

Initializes a newly created NSURL that points to a location specified by resolving bookmark data.

```

- (id)initWithResolvingBookmarkData:(NSData
    *)bookmarkDataoptions:(NSURLBookmarkResolutionOptions)optionsrelativeToURL:(NSURL
    *)relativeURLbookmarkDataIsStale:(BOOL *)isStaleerror:(NSError **)error

```

Parameters*bookmarkData*

The bookmark data the URL is derived from.

options

Options taken into account when resolving the bookmark data.

relativeURL

The base URL that the bookmark data is relative to.

isStale

If YES, the bookmark data is stale.

error

The error that occurred in the case that the URL cannot be created.

Return ValueAn NSURL initialized by resolving *bookmarkData*.**Availability**

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

initWithFileURLWithPath:

Initializes a newly created NSURL referencing the local file or directory at *path*.

```
- (id)initWithFileURLWithPath:(NSString *)path
```

Parameters

path

The path that the NSURL object will represent. *path* should be a valid system path. If *path* begins with a tilde, it must first be expanded with `stringByExpandingTildeInPath`.

Passing `nil` for this parameter produces an exception.

Return Value

An NSURL object initialized with *path*.

Discussion

Invoking this method is equivalent to invoking `initWithScheme:host:path:` (page 23) with scheme `NSURLFileScheme`, a `nil` host, and *path*.

This method examines *path* in the file system to determine if it is a directory. If *path* is a directory, then a trailing slash is appended. If the file does not exist, it is assumed that *path* represents a directory and a trailing slash is appended. As an alternative, consider using `initWithFileURLWithPath:isDirectory:` (page 22) which allows you to explicitly specify whether the returned NSURL represents a file or directory.

Availability

Available in Mac OS X v10.0 and later.

See Also

[fileURLWithPath:](#) (page 12)

Related Sample Code

AttachAScript

CoreRecipes

LSMSmartCategorizer

QuickLookDownloader

Declared In

NSURL.h

initWithFileURLWithPath:isDirectory:

Initializes a newly created NSURL referencing the local file or directory at *path*.

```
- (id)initWithFileURLWithPath:(NSString *)path isDirectory:(BOOL)isDir
```

Parameters

path

The path that the NSURL object will represent. *path* should be a valid system path. If *path* begins with a tilde, it must first be expanded with `stringByExpandingTildeInPath`.

Passing `nil` for this parameter produces an exception.

isDir

A Boolean value that specifies whether *path* is treated as a directory path when resolving against relative path components. Pass `YES` if the *path* indicates a directory, `NO` otherwise

Return Value

An NSURL object initialized with *path*.

Discussion

Invoking this method is equivalent to invoking `initWithScheme:host:path:` (page 23) with scheme `NSFileScheme`, a `nil` *host*, and *path*.

Availability

Available in Mac OS X v10.5 and later.

See Also

[fileURLWithPath:](#) (page 12)

Declared In

NSURL.h

initWithScheme:host:path:

Initializes a newly created NSURL with a specified scheme, host, and path.

```
- (id)initWithScheme:(NSString *)scheme host:(NSString *)host path:(NSString *)path
```

Parameters

scheme

The scheme for the NSURL object.

host

The host for the NSURL object. May be the empty string.

path

The path for the NSURL object. If *path* begins with a tilde, it must first be expanded with `stringByExpandingTildeInPath`.

Return Value

The newly initialized NSURL object.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CoreRecipes

Declared In

NSURL.h

initWithString:

Initializes an NSURL object with a provided string.

```
- (id)initWithString:(NSString *)URLString
```

Parameters*URLString*

The string with which to initialize the NSURL object. Must conform to RFC 2396. This method parses *URLString* according to RFCs 1738 and 1808.

Return Value

An NSURL object initialized with *URLString*. If the string was malformed, returns `nil`.

Discussion

This method expects *URLString* to contain any necessary percent escape codes, which are `'%'`, `'/'`, `'#'`, `'.'`, and `'@'`. Note that `'%'` escapes are translated via UTF-8.

Availability

Available in Mac OS X v10.0 and later.

See Also

[URLWithString:](#) (page 15)

Declared In

NSURL.h

initWithString:relativeToURL:

Initializes an NSURL object with a base URL and a relative string.

```
- (id)initWithString:(NSString *)URLString relativeToURL:(NSURL *)baseURL
```

Parameters*URLString*

The string with which to initialize the NSURL object. Must conform to RFC 2396. *URLString* is interpreted relative to *baseURL*.

baseURL

The base URL for the NSURL object.

Return Value

An NSURL object initialized with *URLString* and *baseURL*. If *URLString* was malformed, returns `nil`.

Discussion

This method expects *URLString* to contain any necessary percent escape codes.

`initWithString:relativeToURL:` is the designated initializer for NSURL.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [baseURL](#) (page 18)

- [relativeString](#) (page 28)

[URLWithString:relativeToURL:](#) (page 16)

Declared In

NSURL.h

isEqual:

Returns a Boolean value that indicates whether the receiver and a given object are equal.

```
- (BOOL)isEqual:(id)anObject
```

Parameters

anObject

The object to be compared to the receiver.

Return Value

YES if the receiver and *anObject* are equal, otherwise NO.

Discussion

This method defines what it means for instances to be equal. Two NSURLs are considered equal if and only if they return identical values for both [baseURL](#) (page 18) and [relativeString](#) (page 28).

isFileReferenceURL

Returns whether the URL is a file reference URL.

```
- (BOOL)isFileReferenceURL
```

Return Value

YES if the URL is a file reference URL; otherwise, NO.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

isFileURL

Returns whether the receiver uses the file scheme.

```
- (BOOL)isFileURL
```

Return Value

Returns YES if the receiver uses the file scheme, NO otherwise.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSURL.h

lastPathComponent

Returns the last path component of a file URL.

```
- (NSString *)lastPathComponent
```

Return Value

The last path component of the URL.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

parameterString

Returns the parameter string of a URL conforming to RFC 1808.

- (NSString *)parameterString

Return Value

The parameter string of the URL. If the receiver does not conform to RFC 1808, returns `nil`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSURL.h

password

Returns the password of a URL conforming to RFC 1808.

- (NSString *)password

Return Value

The password of the URL. If the receiver does not conform to RFC 1808, returns `nil`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSURL.h

path

Returns the path of a URL conforming to RFC 1808.

- (NSString *)path

Return Value

The path of the URL. If the receiver does not conform to RFC 1808, returns `nil`. If `isFileURL` (page 25) returns `YES`, the return value is suitable for input into `NSFileManager` or `NSPathUtilities`. If the path has a trailing slash it is stripped.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CoreRecipes
File Wrappers with Core Data Documents
FunHouse
MovieAssembler
SourceView

Declared In

NSURL.h

pathComponents

Returns the individual path components of a file URL in an array.

- (NSArray *)pathComponents

Return Value

An array containing the individual path components of the URL.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

pathExtension

Returns the path extension of a file URL.

- (NSString *)pathExtension

Return Value

The path extension of the URL.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

port

Returns the port number of a URL conforming to RFC 1808.

- (NSNumber *)port

Return Value

The port number of the URL. If the receiver does not conform to RFC 1808, returns `nil`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSURL.h

query

Returns the query of a URL conforming to RFC 1808.

- (NSString *)query

Return ValueThe query of the URL. If the receiver does not conform to RFC 1808, returns `nil`.**Availability**

Available in Mac OS X v10.0 and later.

Declared In

NSURL.h

relativePath

Returns the path of a URL conforming to RFC 1808, without resolving against the receiver's base URL.

- (NSString *)relativePath

Return ValueThe relative path of the URL without resolving against the base URL. If the receiver is an absolute URL, this method returns the same value as [path](#) (page 26). If the receiver does not conform to RFC 1808, returns `nil`.**Availability**

Available in Mac OS X v10.0 and later.

Related Sample Code

IdentitySample

Declared In

NSURL.h

relativeString

Returns a string representation of the relative portion of the URL.

- (NSString *)relativeString

Return ValueA string representation of the relative portion of the URL. If the receiver is an absolute URL this method returns the same value as [absoluteString](#) (page 17).**Availability**

Available in Mac OS X v10.0 and later.

Declared In

NSURL.h

resourceSpecifier

Returns the resource specifier of the URL.

- (NSString *)resourceSpecifier

Return Value

The resource specifier of the URL.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSURL.h

resourceValuesForKeys:error:

Returns the resource values for the properties identified by specified array of keys.

- (NSDictionary *)resourceValuesForKeys:(NSArray *)keyseerror:(NSError **)error

Parameters*keys*

An array of names of URL resource properties.

error

The error that occurred in the case that one or more resource values cannot be retrieved.

Return Value

A dictionary of resource values indexed by key.

DiscussionIf an error occurs, this method returns *nil*.

A key is left out of the returned dictionary if its corresponding resource value is not defined for the URL.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

scheme

Returns the scheme of the URL.

- (NSString *)scheme

Return Value

The scheme of the URL.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

NewsReader

Declared In

NSURL.h

setResourceValue:forKey:error:

Sets the resource property of the URL specified by a given key to a given value.

```
- (BOOL)setResourceValue:(id)value forKey:(NSString *)key error:(NSError **)error
```

Parameters

value

The value for the resource property defined by *key*.

key

The name of one of the URL's resource properties.

error

The error that occurred in the case that the resource value cannot be set.

Return Value

YES if the resource property named *key* is successfully set to *value*; otherwise, NO.

Discussion

The resource is modified synchronously.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

setResourceValues:error:

Sets resource properties of the URL specified by a given set of keys to a given set of values.

```
- (BOOL)setResourceValues:(NSDictionary *)keyedValues error:(NSError **)error
```

Parameters

keyedValues

A dictionary of resource values to be set.

error

The error that occurred in the case that one or more resource values cannot be set.

Return Value

YES if all resource values in *keyedValues* are successfully set; otherwise, NO.

Discussion

If an error occurs during the execution of this method, *error* will contain an array of the resource values that were not successfully set in its *userInfo* dictionary.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

standardizedURL

Returns a new NSURL object with any instances of "." or "." removed from its path.

- (NSURL *)standardizedURL

Return Value

A new NSURL object initialized with a version of the receiver's URL that has had any instances of "." or "." removed from its path.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSURL.h

URLByAppendingPathComponent:

Returns a new URL made by appending a path component to the original URL.

- (NSURL *)URLByAppendingPathComponent:(NSString *)*pathComponent*

Parameters

pathComponent

The path component to add to the URL.

Return Value

A new URL with *pathComponent* appended.

Discussion

If the original URL does not end with a forward slash and *pathComponent* does not begin with a forward slash, a forward slash is inserted between the two parts of the returned URL, unless the original URL is the empty string.

Availability

Available in Mac OS X v10.6 and later.

Related Sample Code

IconCollection

Declared In

NSURL.h

URLByAppendingPathExtension:

Returns a new URL made by appending a path extension to the original URL.

```
- (NSURL *)URLByAppendingPathExtension:(NSString *)pathExtension
```

Parameters

pathExtension

The path extension to add to the URL.

Return Value

A new URL with `pathExtension` appended.

Discussion

If the original URL ends with one or more forward slashes, these are removed from the returned URL. A period is inserted between the two parts of the new URL.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

URLByDeletingLastPathComponent

Returns a new URL made by deleting the last path component from the original URL.

```
- (NSURL *)URLByDeletingLastPathComponent
```

Return Value

A new URL with the last path component of the original URL removed.

Discussion

If the original URL represents the root path, the returned URL is identical. Otherwise, if the original URL has only one path component, the new URL is the empty string.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

URLByDeletingPathExtension

Returns a new URL made by deleting the path extension, if any, from the original URL.

```
- (NSURL *)URLByDeletingPathExtension
```

Return Value

A new URL with the path extension of the original URL removed.

Discussion

If the original URL represents the root path, the returned URL is identical. If the URL has multiple path extensions, only the last one is removed.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

URLByResolvingSymlinksInPath

Returns a new URL that points to the same resource as the original URL and includes no symbolic links.

- (NSURL *)URLByResolvingSymlinksInPath

Return Value

A new URL that points to the same resource as the original URL and includes no symbolic links.

Discussion

If the original URL has no symbolic links, the returned URL is identical to the original URL.

This method only works on URLs with the `file:` path scheme. This method will return an identical URL for all other URLs.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

URLByStandardizingPath

Returns a new URL that points to the same resource as the original URL and is an absolute path.

- (NSURL *)URLByStandardizingPath

Return Value

A new URL that points to the same resource as the original URL and is an absolute path.

Discussion

This method only works on URLs with the `file:` path scheme. This method will return an identical URL for all other URLs.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSURL.h

user

Returns the user portion of a URL conforming to RFC 1808.

- (NSString *)user

Return Value

The user portion of the URL. If the receiver does not conform to RFC 1808, returns `nil`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSURL.h

Constants

NSURL Schemes

These schemes are the ones that NSURL can parse.

```
NSString * const NSURLFileScheme;
```

Constants

NSURLFileScheme

Identifies a URL that points to a file on a mounted volume.

Available in Mac OS X v10.0 and later.

Declared in NSURL.h.

Discussion

For more information, see [initWithScheme:host:path:](#) (page 23).

Common File System Resource Keys

Keys that are applicable to file system URLs.

```

NSString * const NSURLNameKey;
NSString * const NSURLLocalizedNameKey;
NSString * const NSURLIsRegularFileKey;
NSString * const NSURLIsDirectoryKey;
NSString * const NSURLIsSymbolicLinkKey;
NSString * const NSURLIsVolumeKey;
NSString * const NSURLIsPackageKey;
NSString * const NSURLIsSystemImmutableKey;
NSString * const NSURLIsUserImmutableKey;
NSString * const NSURLIsHiddenKey;
NSString * const NSURLHasHiddenExtensionKey;
NSString * const NSURLCreationDateKey;
NSString * const NSURLContentAccessDateKey;
NSString * const NSURLContentModificationDateKey;
NSString * const NSURLAttributeModificationDateKey;
NSString * const NSURLLinkCountKey;
NSString * const NSURLParentDirectoryURLKey;
NSString * const NSURLVolumeURLKey;
NSString * const NSURLTypeIDentifierKey;
NSString * const NSURLLocalizedTypeDescriptionKey;
NSString * const NSURLLabelNumberKey;
NSString * const NSURLLabelColorKey;
NSString * const NSURLLocalizedLabelKey;
NSString * const NSURLEffectiveIconKey;
NSString * const NSURLCustomIconKey;

```

Constants

NSURLNameKey

Key for the resource's name in the file system, returned as an NSString object.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

NSURLLocalizedNameKey

Key for the resource's localized or extension-hidden name, returned as an NSString object.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

NSURLIsRegularFileKey

Key for determining whether the resource is a regular file, as opposed to a directory or a symbolic link. Returned as an NSNumber object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

NSURLIsDirectoryKey

Key for determining whether the resource is a directory, returned as an NSNumber object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

NSURLIsSymbolicLinkKey

Key for determining whether the resource is a symbolic link, returned as an NSNumber object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

`NSURLIsVolumeKey`

Key for determining whether the resource is the root directory of a volume, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLIsPackageKey`

Key for determining whether the resource is a packaged directory, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLIsSystemImmutableKey`

Key for determining whether the resource's system immutable bit is set, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLIsUserImmutableKey`

Key for determining whether the resource's user immutable bit is set, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLIsHiddenKey`

Key for determining whether the resource is normally not displayed to users, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLHasHiddenExtensionKey`

Key for determining whether the resource's extension is normally removed from its localized name, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLCreationDateKey`

Key for the resource's creation date, returned as an `NSDate` object if the volume supports creation dates, or `nil` if creation dates are unsupported.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLContentAccessDateKey`

Key for the last time the resource was accessed, returned as an `NSDate` object if the volume supports access dates, or `nil` if access dates are unsupported.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLContentModificationDateKey`

Key for the last time the resource was modified, returned as an `NSDate` object if the volume supports modification dates, or `nil` if modification dates are unsupported.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLAttributeModificationDateKey`

Key for the last time the resource's attributes were modified, returned as an `NSDate` object if the volume supports attribute modification dates, or `nil` if attribute modification dates are unsupported.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLLinkCountKey`

Key for the number of hard links to the resource, returned as an `NSNumber` object.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLParentDirectoryURLKey`

Key for the parent directory of the resource, returned as an `NSURL` object, or `nil` if the resource is the root directory of its volume.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLVolumeURLKey`

Key for the root directory of the resource's volume, returned as an `NSURL` object.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLTypeIDentifierKey`

Key for the resource's uniform type identifier (UTI), returned as an `NSString` object.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLLocalizedTypeDescriptionKey`

Key for the resource's localized type description, returned as an `NSString` object.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLLabelNumberKey`

Key for the resource's label number, returned as an `NSNumber` object.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLLabelColorKey`

Key for the resource's label color, returned as an `NSColor` object, or `nil` if the resource has no label color.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLLocalizedLabelKey`

Key for the resource's localized label text, returned as an `NSString` object, or `nil` if the resource has no localized label text.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLEffectiveIconKey`

Key for the resource's normal icon, returned as an `NSImage` object.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLCustomIconKey`

Key for the icon stored with the resource, returned as an `NSImage` object, or `nil` if the resource has no custom icon.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

File Property Keys

Keys that apply to properties of files.

```
NSString * const NSURLFileSizeKey;  
NSString * const NSURLFileAllocatedSizeKey;  
NSString * const NSURLIsAliasFileKey;
```

Constants

`NSURLFileSizeKey`

Key for the file's size in bytes, returned as an `NSNumber` object.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLFileAllocatedSizeKey`

Key for the total size allocated on disk for the file, returned as an `NSNumber` object.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLIsAliasFileKey`

Key for determining whether the file is an alias, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

Volume Property Keys

Keys that apply to volumes.

```

NSString * const NSURLVolumeLocalizedFormatDescriptionKey;
NSString * const NSURLVolumeTotalCapacityKey;
NSString * const NSURLVolumeAvailableCapacityKey;
NSString * const NSURLVolumeResourceCountKey;
NSString * const NSURLVolumeSupportsPersistentIDsKey;
NSString * const NSURLVolumeSupportsSymbolicLinksKey;
NSString * const NSURLVolumeSupportsHardLinksKey;
NSString * const NSURLVolumeSupportsJournalingKey;
NSString * const NSURLVolumeIsJournalingKey;
NSString * const NSURLVolumeSupportsSparseFilesKey;
NSString * const NSURLVolumeSupportsZeroRunsKey;
NSString * const NSURLVolumeSupportsCaseSensitiveNamesKey;
NSString * const NSURLVolumeSupportsCasePreservedNamesKey;

```

Constants

NSURLVolumeLocalizedFormatDescriptionKey

Key for the volume’s descriptive format name, returned as an NSString object.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

NSURLVolumeTotalCapacityKey

Key for the volume’s capacity in bytes, returned as an NSNumber object.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

NSURLVolumeAvailableCapacityKey

Key for the volume’s available capacity in bytes, returned as an NSNumber object.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

NSURLVolumeResourceCountKey

Key for the total number of resources on the volume, returned as an NSNumber object.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

NSURLVolumeSupportsPersistentIDsKey

Key for determining whether the volume supports persistent IDs, returned as an NSNumber object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

NSURLVolumeSupportsSymbolicLinksKey

Key for determining whether the volume supports symbolic links, returned as an NSNumber object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

NSURLVolumeSupportsHardLinksKey

Key for determining whether the volume supports hard links, returned as an NSNumber object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in NSURL.h.

`NSURLVolumeSupportsJournalingKey`

Key for determining whether the volume supports journaling, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLVolumeIsJournalingKey`

Key for determining whether the volume is currently journaling, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLVolumeSupportsSparseFilesKey`

Key for determining whether the volume supports sparse files, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLVolumeSupportsZeroRunsKey`

Key for determining whether the volume supports zero runs, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLVolumeSupportsCaseSensitiveNamesKey`

Key for determining whether the volume supports case-sensitive names, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

`NSURLVolumeSupportsCasePreservedNamesKey`

Key for determining whether the volume supports case-preserved names, returned as an `NSNumber` object with value 0 or 1.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

Bookmark Data Creation Options

Options used when creating bookmark data.

```
enum {
    NSURLBookmarkCreationPreferFileIDResolution = ( 1UL << 8 ),
    NSURLBookmarkCreationMinimalBookmark = ( 1UL << 9 ),
    NSURLBookmarkCreationSuitableForBookmarkFile = ( 1UL << 10 )
};
typedef NSUInteger NSURLBookmarkCreationOptions;
typedef NSUInteger NSURLBookmarkFileCreationOptions;
```

Constants

NSURLBookmarkCreationPreferFileIDResolution

Option for specifying that an alias created with the bookmark data prefers resolving with its embedded file ID.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

NSURLBookmarkCreationMinimalBookmark

Option for specifying that an alias created with the bookmark data be created with minimal information, which may make it smaller but still able to resolve in certain ways.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

NSURLBookmarkCreationSuitableForBookmarkFile

Option for specifying that the bookmark data include properties required to create Finder alias files.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

Bookmark Data Resolution Options

Options used when resolving bookmark data.

```
enum {
    NSURLBookmarkResolutionWithoutUI = ( 1UL << 8 ),
    NSURLBookmarkResolutionWithoutMounting = ( 1UL << 9 )
};
typedef NSUInteger NSURLBookmarkResolutionOptions;
```

Constants

NSURLBookmarkResolutionWithoutUI

Option for specifying that no UI feedback accompany resolution of the bookmark data.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

NSURLBookmarkResolutionWithoutMounting

Option for specifying that no volume should be mounted during resolution of the bookmark data.

Available in Mac OS X v10.6 and later.

Declared in `NSURL.h`.

NSURLHandle FTP Property Keys

FTP-specific property keys.

```

NSString *NSFTPPropertyUserLoginKey;
NSString *NSFTPPropertyUserPasswordKey;
NSString *NSFTPPropertyActiveTransferModeKey;
NSString *NSFTPPropertyFileOffsetKey;
NSString *NSFTPPropertyFTPProxy;

```

Constants

`NSFTPPropertyUserLoginKey`

Key for the user login, returned as an `NSString` object.

The default value for this key is “anonymous”.

Available in Mac OS X v10.2 and later.

Deprecated in Mac OS X v10.4.

Declared in `NSURLHandle.h`.

`NSFTPPropertyUserPasswordKey`

Key for the user password, returned as an `NSString` object.

The default value for this key is “NSURLHandle@apple.com”.

Available in Mac OS X v10.2 and later.

Deprecated in Mac OS X v10.4.

Declared in `NSURLHandle.h`.

`NSFTPPropertyActiveTransferModeKey`

Key for retrieving whether in active transfer mode, returned as a boolean wrapped in an `NSNumber` object.

The default value for this key is NO (passive mode).

Available in Mac OS X v10.2 and later.

Deprecated in Mac OS X v10.4.

Declared in `NSURLHandle.h`.

`NSFTPPropertyFileOffsetKey`

Key for retrieving the file offset, returned as an `NSNumber` object. The default value for this key is zero.

Available in Mac OS X v10.2 and later.

Deprecated in Mac OS X v10.4.

Declared in `NSURLHandle.h`.

`NSFTPPropertyFTPProxy`

`NSDictionary` containing proxy information to use in place of proxy identified in `SystemConfiguration.framework`.

To avoid any proxy use, pass an empty dictionary.

Available in Mac OS X v10.3 and later.

Deprecated in Mac OS X v10.4.

Declared in `NSURLHandle.h`.

Discussion

Pass these keys to `NSURLHandle`’s `propertyForKeyIfAvailable:` to request specific data. All keys are optional. The default configuration allows an anonymous, passive-mode, one-off transfer of the specified URL.

NSURLHandle HTTP Property Keys

HTTP-specific property keys.

```
NSString * const NSHTTPPropertyStatusCodeKey;
NSString * const NSHTTPPropertyStatusReasonKey;
NSString * const NSHTTPPropertyServerHTTPVersionKey;
NSString * const NSHTTPPropertyRedirectionHeadersKey;
NSString * const NSHTTPPropertyErrorPageDataKey;
NSString * const NSHTTPPropertyHTTPProxy;
```

Constants

`NSHTTPPropertyStatusCodeKey`

Key for the status code, returned as an integer wrapped in an `NSNumber` object.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared in `NSURLHandle.h`.

`NSHTTPPropertyStatusReasonKey`

Key for the remainder of the HTTP status line following the status code, returned as an `NSString` object.

This string usually contains an explanation of the error in English. Because this string is taken straight from the server response, it's not localized.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared in `NSURLHandle.h`.

`NSHTTPPropertyServerHTTPVersionKey`

Key for retrieving the HTTP version as an `NSString` object containing the initial server status line up to the first space.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared in `NSURLHandle.h`.

`NSHTTPPropertyRedirectionHeadersKey`

Key for retrieving the redirection headers as an `NSDictionary` object with each header value keyed to the header name.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared in `NSURLHandle.h`.

`NSHTTPPropertyErrorPageDataKey`

Key for retrieving an error page as an `NSData` object.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared in `NSURLHandle.h`.

NSHTTPPropertyHTTPProxy

Key for retrieving the `NSDictionary` object containing proxy information to use in place of proxy identified in `SystemConfiguration.framework`.

To avoid any proxy use, pass an empty dictionary.

Available in Mac OS X v10.2 and later.

Deprecated in Mac OS X v10.4.

Declared in `NSURLHandle.h`.

Discussion

Pass these keys to `NSURLHandle`'s `propertyForKeyIfAvailable:` to request specific data.

Deprecated NSURL Methods

A method identified as deprecated has been superseded and may become unsupported in the future.

Deprecated in Mac OS X v10.4

loadResourceDataNotifyingClient:usingCache:

Loads the receiver's resource data in the background. (Deprecated in Mac OS X v10.4.)

```
- (void)loadResourceDataNotifyingClient:(id)client usingCache:(BOOL)shouldUseCache
```

Parameters

client

The client of the loading operation. *client* is notified of the receiver's progress loading the resource data using the NSURLClient informal protocol. The NSURLClient messages are delivered on the current thread and require the run loop to be running.

shouldUseCache

Whether the URL should use cached resource data from an already loaded URL that refers to the same resource. If *YES*, the cache is consulted when loading data. If *NO*, the data is always loaded directly, without consulting the cache.

Discussion

A given NSURL object can perform only one background load at a time.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

NSURL.h

propertyForKey:

Returns the specified property of the receiver's resource. (Deprecated in Mac OS X v10.4.)

```
- (id)propertyForKey:(NSString *)propertyKey
```

Parameters

propertyKey

The key of the desired property.

Return Value

The value of the property of the receiver's resource for the provided key. Returns `nil` if there is no such key.

Deprecated NSURL Methods

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

See Also

[setProperty:forKey:](#) (page 46)

Declared In

NSURL.h

resourceDataUsingCache:

Returns the receiver's resource data, loading it if necessary. Use `NSURLConnection` instead of this method. (Deprecated in Mac OS X v10.4.)

```
- (NSData *)resourceDataUsingCache:(BOOL)shouldUseCache
```

Parameters

shouldUseCache

Whether the URL should use cached resource data from an already loaded URL that refers to the same resource. If *YES*, the cache is consulted when loading data. If *NO*, the data is always loaded directly, without consulting the cache.

Return Value

The receiver's resource data.

Discussion

If the receiver has not already loaded its resource data, it will attempt to load it as a blocking operation.

In Mac OS X v10.4, this method requests that the data be sent with gzip compression, however it does not automatically decompress the data if the server complies with this request. Data is automatically decompressed in Mac OS X v10.5 and later.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Related Sample Code

ImageClient

Declared In

NSURL.h

setProperty:forKey:

Changes the specified property of the receiver's resource. (Deprecated in Mac OS X v10.4.)

```
- (BOOL)setProperty:(id)propertyValue forKey:(NSString *)propertyKey
```

Parameters

propertyValue

The new value of the property of the receiver's resource.

Deprecated NSURL Methods

propertyKey

The key of the desired property.

Return Value

Returns YES if the modification was successful, NO otherwise.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

NSURL.h

setResourceData:

Attempts to set the resource data for the receiver. (Deprecated in Mac OS X v10.4.)

```
- (BOOL)setResourceData:(NSData *)data
```

Parameters

data

The data to set for the URL.

Return Value

Returns YES if successful, NO otherwise.

Discussion

In the case of a file URL, setting the data involves writing *data* to the specified file.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

NSURL.h

URLHandleUsingCache:

Returns a URL handle to service the receiver. (Deprecated in Mac OS X v10.4.)

```
- (NSURLHandle *)URLHandleUsingCache:(BOOL)shouldUseCache
```

Parameters

shouldUseCache

Whether to use a cached URL handle. If *shouldUseCache* is YES, the cache is searched for a URL handle that has serviced the receiver or another identical URL. If *shouldUseCache* is NO, a newly instantiated handle is returned, even if an equivalent URL has been loaded.

Return Value

A URL handle to service the receiver.

Discussion

Sophisticated clients use the URL handle directly for additional control.

Deprecated NSURL Methods

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

See Also

cachedHandleForURL: (NSURLHandle)

Declared In

NSURL.h

Document Revision History

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

Date	Notes
2009-10-09	Added exception information for file URL initialization methods.
2009-08-28	Update to Mac OS X v10.6. Added bookmark data API.
2009-02-04	Miscellaneous edits.
2008-11-19	Added class specific behavior for isEqual:
2007-02-23	Updated to include new API introduced in Mac OS X v10.5.
2006-05-23	First publication of this content as a separate document.
	First publication of this content as a separate document.

REVISION HISTORY

Document Revision History

Index

A

`absoluteString` instance method [17](#)
`absoluteURL` instance method [18](#)

B

`baseURL` instance method [18](#)
Bookmark Data Creation Options [40](#)
Bookmark Data Resolution Options [41](#)
`bookmarkDataWithContentsOfURL:error:` class method [12](#)
`bookmarkDataWithOptions:`
 `includingResourceValuesForKeys:relativeToURL:`
 `error:` instance method [18](#)

C

`checkResourceIsReachableAndReturnError:`
 instance method [19](#)
Common File System Resource Keys [34](#)

F

File Property Keys [38](#)
`filePathURL` instance method [19](#)
`fileReferenceURL` instance method [19](#)
`fileURLWithPath:` class method [12](#)
`fileURLWithPath:isDirectory:` class method [13](#)
`fileURLWithPathComponents:` class method [14](#)
`fragment` instance method [20](#)

G

`getResourceValue:forKey:error:` instance method [20](#)

H

`host` instance method [21](#)

I

`initWithResolvingBookmarkData:options:relativeToURL:`
 `bookmarkDataIsStale:error:` instance method [21](#)
`initWithFileURLWithPath:` instance method [22](#)
`initWithFileURLWithPath:isDirectory:` instance method [22](#)
`initWithScheme:host:path:` instance method [23](#)
`initWithString:` instance method [23](#)
`initWithString:relativeToURL:` instance method [24](#)
`isEqual:` instance method [25](#)
`isFileReferenceURL` instance method [25](#)
`isFileURL` instance method [25](#)

L

`lastPathComponent` instance method [25](#)
`loadResourceDataNotifyingClient:usingCache:`
 instance method [45](#)

N

`NSFTPPropertyActiveTransferModeKey` constant [42](#)
`NSFTPPropertyFileOffsetKey` constant (Deprecated in Mac OS X v10.4) [42](#)

NSFTPPropertyFTPProxy **constant** 42
 NSFTPPropertyUserLoginKey **constant** 42
 NSFTPPropertyUserPasswordKey **constant** 42
 NSHTTPPropertyErrorPageDataKey **constant**
 (Deprecated in Mac OS X v10.4) 43
 NSHTTPPropertyHTTPProxy **constant** 44
 NSHTTPPropertyRedirectionHeadersKey **constant**
 (Deprecated in Mac OS X v10.4) 43
 NSHTTPPropertyServerHTTPVersionKey **constant**
 (Deprecated in Mac OS X v10.4) 43
 NSHTTPPropertyStatusCodeKey **constant** (Deprecated
 in Mac OS X v10.4) 43
 NSHTTPPropertyStatusReasonKey **constant** 43
NSURL Schemes 34
 NSURLAttributeModificationDateKey **constant** 37
 NSURLBookmarkCreationMinimalBookmark **constant**
 41
 NSURLBookmarkCreationPreferFileIDResolution
constant 41
 NSURLBookmarkCreationSuitableForBookmarkFile
constant 41
 NSURLBookmarkResolutionWithoutMounting
constant 41
 NSURLBookmarkResolutionWithoutUI **constant** 41
 NSURLContentAccessDateKey **constant** 36
 NSURLContentModificationDateKey **constant** 37
 NSURLCreationDateKey **constant** 36
 NSURLCustomIconKey **constant** 38
 NSURLEffectiveIconKey **constant** 38
 NSURLFileAllocatedSizeKey **constant** 38
 NSURLFileScheme **constant** 34
 NSURLFileSizeKey **constant** 38
NSURLHandle FTP Property Keys 41
NSURLHandle HTTP Property Keys 43
 NSURLHasHiddenExtensionKey **constant** 36
 NSURLIsAliasFileKey **constant** 38
 NSURLIsDirectoryKey **constant** 35
 NSURLIsHiddenKey **constant** 36
 NSURLIsPackageKey **constant** 36
 NSURLIsRegularFileKey **constant** 35
 NSURLIsSymbolicLinkKey **constant** 35
 NSURLIsSystemImmutableKey **constant** 36
 NSURLIsUserImmutableKey **constant** 36
 NSURLIsVolumeKey **constant** 36
 NSURLLabelColorKey **constant** 37
 NSURLLabelNumberKey **constant** 37
 NSURLLinkCountKey **constant** 37
 NSURLLocalizedLabelKey **constant** 38
 NSURLLocalizedNameKey **constant** 35
 NSURLLocalizedTypeDescriptionKey **constant** 37
 NSURLNameKey **constant** 35
 NSURLParentDirectoryURLKey **constant** 37
 NSURLTypeIDentifierKey **constant** 37

NSURLVolumeAvailableCapacityKey **constant** 39
 NSURLVolumeIsJournalingKey **constant** 40
 NSURLVolumeLocalizedFormatDescriptionKey
constant 39
 NSURLVolumeResourceCountKey **constant** 39
 NSURLVolumeSupportsCasePreservedNamesKey
constant 40
 NSURLVolumeSupportsCaseSensitiveNamesKey
constant 40
 NSURLVolumeSupportsHardLinksKey **constant** 39
 NSURLVolumeSupportsJournalingKey **constant** 40
 NSURLVolumeSupportsPersistentIDsKey **constant**
 39
 NSURLVolumeSupportsSparseFilesKey **constant** 40
 NSURLVolumeSupportsSymbolicLinksKey **constant**
 39
 NSURLVolumeSupportsZeroRunsKey **constant** 40
 NSURLVolumeTotalCapacityKey **constant** 39
 NSURLVolumeURLKey **constant** 37

P

parameterString **instance method** 26
 password **instance method** 26
 path **instance method** 26
 pathComponents **instance method** 27
 pathExtension **instance method** 27
 port **instance method** 27
 propertyForKey: **instance method** 45

Q

query **instance method** 28

R

relativePath **instance method** 28
 relativeString **instance method** 28
 resourceDataUsingCache: **instance method** 46
 resourceSpecifier **instance method** 29
 resourceValuesForKeys:error: **instance method** 29
 resourceValuesForKeys:fromBookmarkData: **class**
method 14

S

scheme **instance method** 29

setProperty:forKey: **instance method** 46
setResourceData: **instance method** 47
setResourceValue:forKey:error: **instance method**
30
setResourceValues:error: **instance method** 30
standardizedURL **instance method** 31

U

URLByAppendingPathComponent: **instance method** 31
URLByAppendingPathExtension: **instance method** 32
URLByDeletingLastPathComponent **instance method**
32
URLByDeletingPathExtension **instance method** 32
URLByResolvingBookmarkData:options:relativeToURL:
bookmarkDataIsStale:error: **class method** 15
URLByResolvingSymlinksInPath **instance method** 33
URLByStandardizingPath **instance method** 33
URLHandleUsingCache: **instance method** 47
URLWithString: **class method** 15
URLWithString:relativeToURL: **class method** 16
user **instance method** 33

V

Volume Property Keys 38

W

writeBookmarkData:toURL:options:error: **class**
method 16