NSURL Class Reference

Cocoa > Networking

2009-02-04

Contents

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 ImageClient iSpend LSMSmartCategorizer StickiesExample

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 18) 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 will return `nil`, which you must be prepared to handle. If you are creating NSURL objects using file system paths, you should use `fileURLWithPath:` (page 8) or `initFileURLWithPath:` (page 12), 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 informal protocol `NSURLClient` defines a set of methods useful for managing the loading of a URL resource in the background.

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

NSURL is "toll-free bridged" with its Core Foundation counterpart, CFURL. 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:`

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

Tasks

Creating an NSURL

`– initWithScheme:host:path:` (page 14)
Initializes a newly created NSURL with a specified scheme, host, and path.

`+ URLWithString:` (page 10)
Creates and returns an NSURL object initialized with a provided string.

`– initWithString:` (page 14)
Initializes an NSURL object with a provided string.

`+ URLWithString:relativeToURL:` (page 10)
Creates and returns an NSURL object initialized with a base URL and a relative string.

- `initWithString:relativeToURL:` (page 15)
  Initializes an NSURL object with a base URL and a relative string.
+ `fileURLWithPath:isDirectory:` (page 9)
  Initializes and returns a newly created NSURL object as a file URL with a specified path.
+ `fileURLWithPath:` (page 8)
  Initializes and returns a newly created NSURL object as a file URL with a specified path.
- `initFileURLWithPath:isDirectory:` (page 13)
  Initializes a newly created NSURL referencing the local file or directory at *path*.
- `initFileURLWithPath:` (page 12)
  Initializes a newly created NSURL referencing the local file or directory at *path*.

Identifying and Comparing Objects

- `isEqual:` (page 15)
  Returns a Boolean value that indicates whether the receiver and a given object are equal.

Querying an NSURL

- `isFileURL` (page 16)
  Returns whether the receiver uses the file scheme.

Loading the Resource of an NSURL Object

- `loadResourceDataNotifyingClient:usingCache:` (page 25) Deprecated in Mac OS X v10.4
  Loads the receiver's resource data in the background.
- `propertyForKey:` (page 25) Deprecated in Mac OS X v10.4
  Returns the specified property of the receiver's resource.
- `resourceDataUsingCache:` (page 26) Deprecated in Mac OS X v10.4
  Returns the receiver's resource data, loading it if necessary.
- `setProperty:forKey:` (page 26) Deprecated in Mac OS X v10.4
  Changes the specified property of the receiver's resource.
- `setResourceData:` (page 27) Deprecated in Mac OS X v10.4
  Attempts to set the resource data for the receiver.
- `URLHandleUsingCache:` (page 27) Deprecated in Mac OS X v10.4
  Returns a URL handle to service the receiver.

Accessing the Parts of the URL

- `absoluteString` (page 11)
  Returns the string for the receiver as if it were an absolute URL.
- `absoluteURL` (page 11)
  Returns an absolute URL that refers to the same resource as the receiver.

– `baseURL` (page 12)
    Returns the base URL of the receiver.

– `fragment` (page 12)
    Returns the fragment of a URL conforming to RFC 1808.

– `host` (page 12)
    Returns the host of a URL conforming to RFC 1808.

– `parameterString` (page 16)
    Returns the parameter string of a URL conforming to RFC 1808.

– `password` (page 16)
    Returns the password of a URL conforming to RFC 1808.

– `path` (page 17)
    Returns the path of a URL conforming to RFC 1808.

– `port` (page 17)
    Returns the port number of a URL conforming to RFC 1808.

– `query` (page 17)
    Returns the query of a URL conforming to RFC 1808.

– `relativePath` (page 18)
    Returns the path of a URL conforming to RFC 1808, without resolving against the receiver's base URL.

– `relativeString` (page 18)
    Returns a string representation of the relative portion of the URL.

– `resourceSpecifier` (page 18)
    Returns the resource specifier of the URL.

– `scheme` (page 19)
    Returns the scheme of the URL.

– `standardizedURL` (page 19)
    Returns a new NSURL object with any instances of `".."` or `"."` removed from its path.

– `user` (page 19)
    Returns the user portion of a URL conforming to RFC 1808.

Class Methods

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`.

Return Value
An NSURL object initialized with *`path`*.

Discussion

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 `fileURLWithPath:isDirectory:` (page 9) 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

`initFileURLWithPath:` (page 12)

Related Sample Code

CoreRecipes

iSpend

Quartz Composer WWDC 2005 TextEdit

StickiesExample

TextEditPlus

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`.

`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

`initFileURLWithPath:` (page 12)

Related Sample Code

AutoSample

IKSlideshowDemo

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

Core Data HTML Store

LSMSmartCategorizer

NewsReader

ObjectPath

VertexPerformanceTest

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
CocoaSOAP
Quartz Composer WWDC 2005 TextEdit
Reducer
TextEditPlus

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
CocoaDragAndDrop
CoreRecipes
NewsReader
Reducer

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.

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`

host

Returns the host of a URL conforming to RFC 1808.

```
- (NSString *)host
```

Return Value
The 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`

initFileURLWithPath:

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

```
- (id)initFileURLWithPath:(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`.

Return Value

An NSURL object initialized with *path*.

Discussion

Invoking this method is equivalent to invoking `initWithScheme:host:path:` (page 14) with scheme `NSFileScheme`, 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 `initFileURLWithPath:isDirectory:` (page 13) 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 8)

Related Sample Code

AttachAScript

CoreRecipes

LSMSmartCategorizer

Quartz Composer WWDC 2005 TextEdit

TextEditPlus

Declared In

`NSURL.h`

initFileURLWithPath:isDirectory:

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

```
- (id)initFileURLWithPath:(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`.

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 14) with scheme `NSFileScheme`, a `nil` host, and *`path`*.

Availability
Available in Mac OS X v10.5 and later.

See Also
`fileURLWithPath:` (page 8)

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 10)

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 12)
`– relativeString` (page 18)
`URLWithString:relativeToURL:` (page 10)

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. For example, two NSURLs are considered equal if they both have the same base `baseURL` (page 12) and `relativeString` (page 18).

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`

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 16) 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
iSpend
QTKitCreateMovie
Quartz Composer WWDC 2005 TextEdit

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 Value
The 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 Value
The 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 17). 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 Value
A 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 11).

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`

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`

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`

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.

```
extern NSString *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 14).

Declared In
`NSURL.h`

NSURLHandle FTP Property Keys

FTP-specific property keys.

```
extern NSString *NSFTPPropertyUserLoginKey;
extern NSString *NSFTPPropertyUserPasswordKey;
extern NSString *NSFTPPropertyActiveTransferModeKey;
extern NSString *NSFTPPropertyFileOffsetKey;
extern 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.

Declared In

`NSURL.h`

NSURLHandle HTTP Property Keys

HTTP-specific property keys.

```
extern NSString *NSHTTPPropertyStatusCodeKey;
extern NSString *NSHTTPPropertyStatusReasonKey;
extern NSString *NSHTTPPropertyServerHTTPVersionKey;
extern NSString *NSHTTPPropertyRedirectionHeadersKey;
extern NSString *NSHTTPPropertyErrorPageDataKey;
extern NSString *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.

Declared In
`NSURL.h`

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.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

See Also
`setProperty:forKey:` (page 26)

Declared In
`NSURL.h`

resourceDataUsingCache:

Returns the receiver's resource data, loading it if necessary. (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.

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.

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.

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
2009-02-04	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.

Index