CFURL Reference

Core Foundation

2009-02-04

Contents

Listings

CFURL Reference

Derived From:	CFType
Framework:	CoreFoundation/CoreFoundation.h
Declared in	CFURL.h

Overview

`CFURL` provides facilities for creating, parsing, and dereferencing URL strings. `CFURL` is useful to applications that need to use URLs to access resources, including local files.

A `CFURL` object is composed of two parts—a base URL, which can be `NULL`, and a string that is resolved relative to the base URL. A `CFURL` object whose string is fully resolved without a base URL is considered absolute; all others are considered relative.

`CFURL` fails to create an object if the string passed is not well-formed (that is, if it does not comply with RFC 2396). Examples of cases that will not succeed are strings containing space characters and high-bit characters. If a function fails to create a `CFURL` object, it returns *`NULL`*, which you must be prepared to handle. If you create `CFURL` objects using file system paths, you should use the `CFURLCreateFromFileSystemRepresentation` (page 23) and `CFURLCreateFromFileSystemRepresentationRelativeToBase` (page 24) functions, which handle the subtle differences between URL paths and file system paths.

For functions that read and write data from a URL, see *Core Foundation URL Access Utilities Reference*

`CFURL` is “toll-free bridged” with its Cocoa Foundation counterpart, NSURL. This means that the Core Foundation type is interchangeable in function or method calls with the bridged Foundation object. In other words, in a method where you see an `NSURL *` parameter, you can pass in a `CFURLRef`, and in a function where you see a `CFURLRef` parameter, you can pass in an `NSURL` instance. This also applies to concrete subclasses of `NSURL`. See Integrating Carbon and Cocoa in Your Application for more information on toll-free bridging.

Functions by Task

Creating a CFURL

`CFURLCopyAbsoluteURL` (page 10)
    Creates a new `CFURL` object by resolving the relative portion of a URL against its base.

`CFURLCreateAbsoluteURLWithBytes` (page 19)
: Creates a new `CFURL` object by resolving the relative portion of a URL, specified as bytes, against its given base URL.

`CFURLCreateCopyAppendingPathComponent` (page 20)
: Creates a copy of a given URL and appends a path component.

`CFURLCreateCopyAppendingPathExtension` (page 21)
: Creates a copy of a given URL and appends a path extension.

`CFURLCreateCopyDeletingLastPathComponent` (page 21)
: Creates a copy of a given URL with the last path component deleted.

`CFURLCreateCopyDeletingPathExtension` (page 22)
: Creates a copy of a given URL with its last path extension removed.

`CFURLCreateFromFileSystemRepresentation` (page 23)
: Creates a new `CFURL` object for a file system entity using the native representation.

`CFURLCreateFromFileSystemRepresentationRelativeToBase` (page 24)
: Creates a `CFURL` object from a native character string path relative to a base URL.

`CFURLCreateFromFSRef` (page 25)
: Creates a URL from a given directory or file.

`CFURLCreateWithBytes` (page 28)
: Creates a `CFURL` object using a given character bytes.

`CFURLCreateWithFileSystemPath` (page 29)
: Creates a `CFURL` object using a local file system path string.

`CFURLCreateWithFileSystemPathRelativeToBase` (page 30)
: Creates a `CFURL` object using a local file system path string relative to a base URL.

`CFURLCreateWithString` (page 31)
: Creates a `CFURL` object using a given `CFString` object.

Accessing the Parts of a URL

`CFURLCanBeDecomposed` (page 10)
: Determines if the given URL conforms to RFC 1808 and therefore can be decomposed.

`CFURLCopyFileSystemPath` (page 11)
: Returns the path portion of a given URL.

`CFURLCopyFragment` (page 11)
: Returns the fragment from a given URL.

`CFURLCopyHostName` (page 12)
: Returns the host name of a given URL.

`CFURLCopyLastPathComponent` (page 13)
: Returns the last path component of a given URL.

`CFURLCopyNetLocation` (page 13)
: Returns the net location portion of a given URL.

`CFURLCopyParameterString` (page 14)
: Returns the parameter string from a given URL.

`CFURLCopyPassword` (page 14)
: Returns the password of a given URL.

`CFURLCopyPath` (page 15)
: Returns the path portion of a given URL.

`CFURLCopyPathExtension` (page 16)
: Returns the path extension of a given URL.

`CFURLCopyQueryString` (page 16)
: Returns the query string of a given URL.

`CFURLCopyResourceSpecifier` (page 17)
: Returns any additional resource specifiers after the path.

`CFURLCopyScheme` (page 17)
: Returns the scheme portion of a given URL.

`CFURLCopyStrictPath` (page 18)
: Returns the path portion of a given URL.

`CFURLCopyUserName` (page 18)
: Returns the user name from a given URL.

`CFURLGetPortNumber` (page 35)
: Returns the port number from a given URL.

`CFURLHasDirectoryPath` (page 36)
: Determines if a given URL's path represents a directory.

Converting URLs to Other Representations

`CFURLCreateData` (page 22)
: Creates a `CFData` object containing the content of a given URL.

`CFURLCreateStringByAddingPercentEscapes` (page 25)
: Creates a copy of a string, replacing certain characters with the equivalent percent escape sequence based on the specified encoding.

`CFURLCreateStringByReplacingPercentEscapes` (page 27)
: Creates a new string by replacing any percent escape sequences with their character equivalent.

`CFURLCreateStringByReplacingPercentEscapesUsingEncoding` (page 27)
: Creates a new string by replacing any percent escape sequences with their character equivalent.

`CFURLGetFileSystemRepresentation` (page 33)
: Fills a buffer with the file system's native string representation of a given URL's path.

`CFURLGetFSRef` (page 34)
: Converts a given URL to a file or directory object.

`CFURLGetString` (page 35)
: Returns the URL as a `CFString` object.

Getting URL Properties

`CFURLGetBaseURL` (page 32)
: Returns the base URL of a given URL if it exists.

`CFURLGetBytes` (page 33)
: Returns by reference the byte representation of a URL object.

`CFURLGetByteRangeForComponent` (page 32)
: Returns the range of the specified component in the bytes of a URL.

`CFURLGetTypeID` (page 36)
: Returns the type identifier for the `CFURL` opaque type.

Functions

CFURLCanBeDecomposed

Determines if the given URL conforms to RFC 1808 and therefore can be decomposed.

```
Boolean CFURLCanBeDecomposed (
   CFURLRef anURL
);
```

Parameters

`anURL`
: The `CFURL` object to test.

Return Value
`true` if *`anURL`* conforms to RFC 1808, `false` otherwise.

Discussion
If a `CFURL` object can be decomposed, you can retrieve separately each of the four components (scheme, net location, path, and resource specifier), as well as the base URL.

Relative URLs are permitted to have only paths (or a variety of other configurations); these are considered decomposable if their base URL is decomposable. If no base URL is present, they are considered decomposable.

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLCopyAbsoluteURL

Creates a new `CFURL` object by resolving the relative portion of a URL against its base.

```
CFURLRef CFURLCopyAbsoluteURL (
   CFURLRef relativeURL
);
```

Parameters

`relativeURL`
: The `CFURL` object to resolve.

Return Value
A new `CFURL` object, or `NULL` if *`relativeURL`* cannot be made absolute. Ownership follows the Create Rule.

Availability
Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
ImageClient

Declared In
`CFURL.h`

CFURLCopyFileSystemPath

Returns the path portion of a given URL.

```
CFStringRef CFURLCopyFileSystemPath (
   CFURLRef anURL,
   CFURLPathStyle pathStyle
);
```

Parameters

`anURL`
: The `CFURL` object whose path you want to obtain.

`pathStyle`
: The operating system path style to be used to create the path. See Path Style (page 39) for a list of possible values.

Return Value
The URL's path in the format specified by *`pathStyle`*. Ownership follows the Create Rule.

Discussion
This function returns the URL's path as a file system path for a given path style.

Availability
Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
AudioBurn

MoreAppleEvents

MoreIsBetter

QISA

SeeMyFriends

Declared In
`CFURL.h`

CFURLCopyFragment

Returns the fragment from a given URL.

```
CFStringRef CFURLCopyFragment (
   CFURLRef anURL,
   CFStringRef charactersToLeaveEscaped
);
```

Parameters

anURL

The `CFURL` object whose fragment you want to obtain.

charactersToLeaveEscaped

Characters whose percent escape sequences, such as `%20` for a space character, you want to leave intact. Pass `NULL` to specify that no percent escapes be replaced, or the empty string (`CFSTR("")`) to specify that all be replaced.

Return Value

The fragment, or `NULL` if no fragment exists. Ownership follows the Create Rule.

Discussion

A fragment is the text following a "#". These are generally used to indicate locations within a single file. This function removes all percent escape sequences except those for characters specified in *charactersToLeaveEscaped*.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Declared In

`CFURL.h`

CFURLCopyHostName

Returns the host name of a given URL.

```
CFStringRef CFURLCopyHostName (
   CFURLRef anURL
);
```

Parameters

anURL

The `CFURL` object to examine.

Return Value

The host name of *anURL*. Ownership follows the Create Rule.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

ImageClient

Declared In

`CFURL.h`

CFURLCopyLastPathComponent

Returns the last path component of a given URL.

```
CFStringRef CFURLCopyLastPathComponent (
   CFURLRef url
);
```

Parameters

url

The `CFURL` object to examine.

Return Value

The last path component of *url*. Ownership follows the Create Rule.

Discussion

Note that if there is no last path component, this function returns an empty string. In the code sample shown in Listing 1, `lastPathComponent` is an empty string.

Listing 1 Code sample illustrating CFURLCopyLastPathComponent

```
CFStringRef urlString = CFSTR("http://www.apple.com");
CFURLRef url = CFURLCreateWithString(NULL, urlString, NULL);
CFStringRef lastPathComponent = CFURLCopyLastPathComponent (url);
```

If `urlString` were created with `CFSTR("http://www.apple.com/")`, then `lastPathComponent` would be a `CFString` object containing the character "/".

See also `CFURLCopyPathExtension` (page 16).

Availability

Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

CFFTPSample

HITextViewDemo

QTCarbonShell

RecentItems

SampleCMPlugIn

Declared In

`CFURL.h`

CFURLCopyNetLocation

Returns the net location portion of a given URL.

```
CFStringRef CFURLCopyNetLocation (
   CFURLRef anURL
);
```

Parameters

anURL

The `CFURL` object to examine.

Return Value
The net location of *anURL*, or `NULL` if the URL cannot be decomposed (doesn't conform to RFC 1808). Ownership follows the Create Rule.

Discussion
The URL net location is the portion of the URL that identifies the network address of the resource. It includes the optional username and password, as well as the target machine's IP address or host name.

This function leaves any percent escape sequences intact.

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLCopyParameterString

Returns the parameter string from a given URL.

```
CFStringRef CFURLCopyParameterString (
   CFURLRef anURL,
   CFStringRef charactersToLeaveEscaped
);
```

Parameters

anURL
The `CFURL` object to examine.

charactersToLeaveEscaped
Characters whose percent escape sequences, such as `%20` for a space character, you want to leave intact. Pass `NULL` to specify that no percent escapes be replaced, or the empty string (`CFSTR("")`) to specify that all be replaced.

Return Value
The parameter string (as defined in RFC 1738), or `NULL` if no parameter string exists. Ownership follows the Create Rule.

Discussion
This function removes all percent escape sequences except those for characters specified in *charactersToLeaveEscaped*.

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLCopyPassword

Returns the password of a given URL.

```
CFStringRef CFURLCopyPassword (
   CFURLRef anURL
);
```

Parameters

anURL

The `CFURL` object to examine.

Return Value

The password, or `NULL` if no password exists. In some cases, this function may also return the empty string (`CFSTR("")`) if no password exists. You should consider `NULL` and the empty string to be equivalent. Ownership follows the Create Rule.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Declared In

`CFURL.h`

CFURLCopyPath

Returns the path portion of a given URL.

```
CFStringRef CFURLCopyPath (
   CFURLRef anURL
);
```

Parameters

anURL

The `CFURL` object to examine.

Return Value

The path of *anURL*, or `NULL` if the URL cannot be decomposed (doesn't conform to RFC 1808). Ownership follows the Create Rule.

Discussion

This function does not resolve the URL against its base and replaces all percent escape sequences. This function's return value includes any leading slash (giving the path the normal POSIX appearance), if present. If this behavior is not appropriate, use `CFURLCopyStrictPath` (page 18) whose return value omits any leading slash. You may also want to use the function `CFURLCopyFileSystemPath` (page 11), which returns the URL's path as a file system path for the given path style. If the path is to be passed to file system calls, you may also want to use the function `CFURLGetFileSystemRepresentation` (page 33), which returns a C string.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

ImageClient

iTunesController

SampleDS

Declared In
`CFURL.h`

CFURLCopyPathExtension

Returns the path extension of a given URL.

```
CFStringRef CFURLCopyPathExtension (
   CFURLRef url
);
```

Parameters

`url`
: The `CFURL` object to examine.

Return Value
The path extension of *`url`*, or `NULL` if no extension exists. Ownership follows the Create Rule.

Discussion
The path extension is the portion of the last path component which follows the final period, if there is one. For example, for `http:/www.apple.com/developer/macosx.today.html`, the extension is `html`, and for `http:/www.apple.com/developer`, there is no path extension.

See also `CFURLCopyLastPathComponent` (page 13).

Availability
Available in CarbonLib v1.1 and later.
Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLCopyQueryString

Returns the query string of a given URL.

```
CFStringRef CFURLCopyQueryString (
   CFURLRef anURL,
   CFStringRef charactersToLeaveEscaped
);
```

Parameters

`anURL`
: The `CFURL` object to examine.

`charactersToLeaveEscaped`
: Characters whose percent escape sequences, such as `%20` for a space character, you want to leave intact. Pass `NULL` to specify that no percent escapes be replaced, or the empty string (`CFSTR("")`) to specify that all be replaced.

Return Value
The query string, or `NULL` if no parameter string exists. Ownership follows the Create Rule.

Discussion
This function removes all percent escape sequences except those for characters specified in *charactersToLeaveEscaped*.

Availability
Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLCopyResourceSpecifier

Returns any additional resource specifiers after the path.

```
CFStringRef CFURLCopyResourceSpecifier (
   CFURLRef anURL
);
```

Parameters
anURL

The `CFURL` object to examine.

Return Value
The resource specifiers. Ownership follows the Create Rule.

Discussion
This function leaves any percent escape sequences intact. For decomposable URLs, this function returns everything after the path. For URLs that cannot be decomposed, this function returns everything except the scheme itself.

Availability
Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLCopyScheme

Returns the scheme portion of a given URL.

```
CFStringRef CFURLCopyScheme (
   CFURLRef anURL
);
```

Parameters
anURL

The `CFURL` object to examine.

Return Value
The scheme of *anURL*. Ownership follows the Create Rule.

Discussion
The URL scheme is the portion of the URL specifying the transport type. For example `http`, `ftp`, and `rtsp` are schemes. This function leaves any percent escape sequences intact.

Availability
Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
ImageClient

Declared In
`CFURL.h`

CFURLCopyStrictPath

Returns the path portion of a given URL.

```
CFStringRef CFURLCopyStrictPath (
   CFURLRef anURL,
   Boolean *isAbsolute
);
```

Parameters
anURL

The `CFURL` object to examine.

isAbsolute

On return, indicates whether the path of *anURL* is absolute.

Return Value
The path of *anURL*, or `NULL` if the URL cannot be decomposed (doesn't conform to RFC 1808). Ownership follows the Create Rule.

Discussion
This function does not resolve the URL against its base and replaces all percent escape sequences. This function's return value does not include a leading slash and uses *isAbsolute* to report whether the URL's path is absolute. If this behavior is not appropriate, use the `CFURLCopyPath` (page 15) function whose return value includes the leading slash (giving the path the normal POSIX appearance). You may also want to use the `CFURLCopyFileSystemPath` (page 11) function, which returns the URL's path as a file system path for the given path style. If the path is to be passed to file system calls, you may also want to use the function `CFURLGetFileSystemRepresentation` (page 33), which returns a C string.

Availability
Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLCopyUserName

Returns the user name from a given URL.

```
CFStringRef CFURLCopyUserName (
   CFURLRef anURL
);
```

Parameters

anURL

The `CFURL` object to examine.

Return Value

The user name, or `NULL` if no user name exists. In some cases, this function may also return the empty string (`CFSTR("")`) if no username exists. You should consider `NULL` and the empty string to be equivalent. Ownership follows the Create Rule.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Declared In

`CFURL.h`

CFURLCreateAbsoluteURLWithBytes

Creates a new `CFURL` object by resolving the relative portion of a URL, specified as bytes, against its given base URL.

```
CFURLRef CFURLCreateAbsoluteURLWithBytes (
   CFAllocatorRef alloc,
   const UInt8 *relativeURLBytes,
   CFIndex length,
   CFStringEncoding encoding,
   CFURLRef baseURL,
   Boolean useCompatibilityMode
);
```

Parameters

allocator

The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

relativeURLBytes

The character bytes that represent a relative URL to convert into a `CFURL` object.

length

The number of bytes in *relativeURLBytes*.

encoding

The string encoding of the `relativeURLBytes` string. This encoding is also used to interpret percent escape sequences.

baseURL

The URL to which *relativeURLBytes* is relative.

useCompatibilityMode

If `true`, the rules historically used on the web are used to resolve the string specified by the *relativeURLBytes* parameter against *baseURL*. These rules are generally listed in the RFC as optional or alternate interpretations. Otherwise, the strict rules from the RFC are used.

Return Value
A new `CFURL` object, or `NULL` if *`relativeURLBytes`* cannot be made absolute. Ownership follows the Create Rule.

Availability
Available in Mac OS X v10.3 and later.

Declared In
`CFURL.h`

CFURLCreateCopyAppendingPathComponent

Creates a copy of a given URL and appends a path component.

```
CFURLRef CFURLCreateCopyAppendingPathComponent (
   CFAllocatorRef allocator,
   CFURLRef url,
   CFStringRef pathComponent,
   Boolean isDirectory
);
```

Parameters

`allocator`
: The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

`url`
: The `CFURL` object to which to append a path component.

`pathComponent`
: The path component to append to *`url`*.

`isDirectory`
: A Boolean value that specifies whether the string is treated as a directory path when resolving against relative path components. Pass `true` if the new component indicates a directory, `false` otherwise.

Return Value
A copy of *`url`* appended with *`pathComponent`*. Ownership follows the Create Rule.

Discussion
The *`isDirectory`* argument specifies whether or not the new path component points to a file or a to directory. Note that the URL syntax for a directory and for a file at otherwise the same location are slightly different—directory URLs must end in "/". If you have the URL `http://www.apple.com/foo/` and you append the path component `bar`, then if *`isDirectory`* is `YES` then the resulting URL is `http://www.apple.com/foo/bar/`, whereas if *`isDirectory`* is `NO` then the resulting URL is `http://www.apple.com/foo/bar`. This difference is particularly important if you resolve another URL against this new URL. `file.html` relative to `http://www.apple.com/foo/bar` is `http://www.apple.com/foo/file.html`, whereas `file.html` relative to `http://www.apple.com/foo/bar/` is `http://www.apple.com/foo/bar/file.html`.

Availability
Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
MoreIsBetter

PDEProject

QISA

simpleJavaLauncher

SpellingChecker-CarbonCocoa

Declared In
`CFURL.h`

CFURLCreateCopyAppendingPathExtension

Creates a copy of a given URL and appends a path extension.

```
CFURLRef CFURLCreateCopyAppendingPathExtension (
   CFAllocatorRef allocator,
   CFURLRef url,
   CFStringRef extension
);
```

Parameters

`allocator`
: The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

`url`
: The `CFURL` object to which to append a path extension.

`extension`
: The extension to append to *`url`*.

Return Value
A copy of *`url`* appended with *`extension`*. Ownership follows the Create Rule.

Availability
Available in CarbonLib v1.1 and later.
Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLCreateCopyDeletingLastPathComponent

Creates a copy of a given URL with the last path component deleted.

```
CFURLRef CFURLCreateCopyDeletingLastPathComponent (
   CFAllocatorRef allocator,
   CFURLRef url
);
```

Parameters

`allocator`
: The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

url

The `CFURL` object whose last path component you want to delete.

Return Value
A copy of *url* with the last path component deleted. Ownership follows the Create Rule.

Availability
Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
HID Utilities Source

ImageClient

Declared In
`CFURL.h`

CFURLCreateCopyDeletingPathExtension

Creates a copy of a given URL with its last path extension removed.

```
CFURLRef CFURLCreateCopyDeletingPathExtension (
   CFAllocatorRef allocator,
   CFURLRef url
);
```

Parameters

allocator

The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

url

The `CFURL` object whose path extension you want to delete.

Return Value
A copy of *url* with its last path extension removed. Ownership follows the Create Rule.

Availability
Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLCreateData

Creates a `CFData` object containing the content of a given URL.

```
CFDataRef CFURLCreateData (
   CFAllocatorRef allocator,
   CFURLRef url,
   CFStringEncoding encoding,
   Boolean escapeWhitespace
);
```

Parameters

`allocator`

The allocator to use to allocate memory for the new `CFData` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

`url`

The URL to convert into a `CFData` object.

`encoding`

The string encoding to use when converting *`url`* into a `CFData` object.

`escapeWhitespace`

`true` if you want to escape whitespace characters in the URL, `false` otherwise.

Return Value
A new `CFData` object containing the content of *`url`*. Ownership follows the Create Rule.

Discussion
This function escapes any character that is not 7-bit ASCII with the byte-code for the given encoding. If *`escapeWhitespace`* is `true`, whitespace characters (' ', '\t', '\r', '\n') will be escaped as well. This is desirable if you want to embed the URL into a larger text stream like HTML.

Availability
Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLCreateFromFileSystemRepresentation

Creates a new `CFURL` object for a file system entity using the native representation.

```
CFURLRef CFURLCreateFromFileSystemRepresentation (
   CFAllocatorRef allocator,
   const UInt8 *buffer,
   CFIndex bufLen,
   Boolean isDirectory
);
```

Parameters

`allocator`

The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

`buffer`

The character bytes to convert into a `CFURL` object. This should be the path as you would use in POSIX function calls.

bufLen

The number of bytes in the buffer.

isDirectory

A Boolean value that specifies whether the string is treated as a directory path when resolving against relative path components—`true` if the pathname indicates a directory, `false` otherwise.

Return Value

A new `CFURL` object. Ownership follows the Create Rule.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

AudioQueueTools

CFFTPSample

CFPrefTopScores

MemoryBasedBundle

MoreIsBetter

Declared In

`CFURL.h`

CFURLCreateFromFileSystemRepresentationRelativeToBase

Creates a `CFURL` object from a native character string path relative to a base URL.

```
CFURLRef CFURLCreateFromFileSystemRepresentationRelativeToBase (
   CFAllocatorRef allocator,
   const UInt8 *buffer,
   CFIndex bufLen,
   Boolean isDirectory,
   CFURLRef baseURL
);
```

Parameters

allocator

The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

buffer

The character bytes to convert into a `CFURL` object. This should be the path as you would use in POSIX function calls.

bufLen

The number of bytes in the buffer.

isDirectory

A Boolean value that specifies whether the string is treated as a directory path when resolving against relative path components. Pass `true` if the pathname indicates a directory, `false` otherwise.

baseURL

The URL against which to resolve the path.

Return Value

A new `CFURL` object. Ownership follows the Create Rule.

Discussion
This function takes a path name in the form of a native character string, resolves it against a base URL, and returns a new `CFURL` object containing the result.

Availability
Available in CarbonLib v1.1 and later.
Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLCreateFromFSRef

Creates a URL from a given directory or file.

```
CFURLRef CFURLCreateFromFSRef (
   CFAllocatorRef allocator,
   const struct FSRef *fsRef
);
```

Parameters

`allocator`
    The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

`fsRef`
    The file or directory representing the URL.

Return Value
A new `CFURL` object. Ownership follows the Create Rule.

Availability
Available in CarbonLib v1.1 and later.
Available in Mac OS X v10.0 and later.

Related Sample Code
BSDLLCTest
CarbonSketch
QTCarbonShell

Declared In
`CFURL.h`

CFURLCreateStringByAddingPercentEscapes

Creates a copy of a string, replacing certain characters with the equivalent percent escape sequence based on the specified encoding.

```
CFStringRef CFURLCreateStringByAddingPercentEscapes (
   CFAllocatorRef allocator,
   CFStringRef originalString,
   CFStringRef charactersToLeaveUnescaped,
   CFStringRef legalURLCharactersToBeEscaped,
   CFStringEncoding encoding
);
```

Parameters

allocator

The allocator to use to allocate memory for the new `CFString` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

originalString

The `CFString` object to copy.

charactersToLeaveUnescaped

Characters whose percent escape sequences you want to leave intact. Pass `NULL` to specify that all escape sequences be replaced.

legalURLCharactersToBeEscaped

Legal characters to be escaped. Pass `NULL` to specify that no legal characters be replaced.

encoding

The encoding to use for the translation. If you are uncertain of the correct encoding, you should use UTF-8, which is the encoding designated by RFC 2396 as the correct encoding for use in URLs.

Return Value

A copy of *originalString* replacing certain characters. If it does not need to be modified (no percent escape sequences are missing), this function may merely return *originalString* with its reference count incremented. Ownership follows the Create Rule.

Discussion

The characters escaped are all characters that are not legal URL characters (based on RFC 2396), plus any characters in *legalURLCharactersToBeEscaped*, less any characters in *charactersToLeaveUnescaped*. To simply correct any non-URL characters in an otherwise correct URL string, pass `NULL` for the *allocator*, *charactersToLeaveEscaped*, and *legalURLCharactersToBeEscaped* parameters, and `kCFStringEncodingUTF8` as the *encoding* parameter.

It may be difficult to use this function to "clean up" unescaped or partially escaped URL strings where sequences are unpredictable and you cannot specify *charactersToLeaveUnescaped*. Instead, you can "pre-process" a URL string using CFURLCreateStringByReplacingPercentEscapesUsingEncoding (page 27) then add the escape characters using CFURLCreateStringByAddingPercentEscapes (page 25), as shown in the following code fragment.

```
CFStringRef originalURLString =
CFSTR("http://online.store.com/storefront/?request=get-document&doi=10.1175%2F1520-0426(2005)014%3C1157:DODADSS%3E2.0.CO%3B2");
CFStringRef preprocessedString =
    CFURLCreateStringByReplacingPercentEscapesUsingEncoding(kCFAllocatorDefault,
 originalURLString, CFSTR(""), kCFStringEncodingUTF8);
CFStringRef urlString =
    CFURLCreateStringByAddingPercentEscapes(kCFAllocatorDefault,
preprocessedString, NULL, NULL, kCFStringEncodingUTF8);
url = CFURLCreateWithString(kCFAllocatorDefault, urlString, NULL);
```

Availability

Available in CarbonLib v1.3 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
CFNetworkHTTPDownload

Declared In
`CFURL.h`

CFURLCreateStringByReplacingPercentEscapes

Creates a new string by replacing any percent escape sequences with their character equivalent.

```
CFStringRef CFURLCreateStringByReplacingPercentEscapes (
   CFAllocatorRef allocator,
   CFStringRef originalString,
   CFStringRef charactersToLeaveEscaped
);
```

Parameters

allocator
The allocator to use to allocate memory for the new `CFString` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

originalString
The `CFString` object to be copied and modified.

charactersToLeaveEscaped
Characters whose percent escape sequences, such as `%20` for a space character, you want to leave intact. Pass `NULL` to specify that no percent escapes be replaced, or the empty string (`CFSTR("")`) to specify that all be replaced.

Return Value
A new `CFString` object, or `NULL` if the percent escapes cannot be converted to characters, assuming UTF-8 encoding. If no characters need to be replaced, this function returns the original string with its reference count incremented. Ownership follows the Create Rule.

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Related Sample Code
CFNetworkHTTPDownload

Declared In
`CFURL.h`

CFURLCreateStringByReplacingPercentEscapesUsingEncoding

Creates a new string by replacing any percent escape sequences with their character equivalent.

```
CFStringRef CFURLCreateStringByReplacingPercentEscapesUsingEncoding (
   CFAllocatorRef allocator,
   CFStringRef origString,
   CFStringRef charsToLeaveEscaped,
   CFStringEncoding encoding
);
```

Parameters

allocator

The allocator to use to allocate memory for the new `CFString` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

originalString

The `CFString` object to be copied and modified.

charactersToLeaveEscaped

Characters whose percent escape sequences, such as `%20` for a space character, you want to leave intact. Pass `NULL` to specify that no percent escapes be replaced, or the empty string (`CFSTR("")`) to specify that all be replaced.

encoding

Specifies the encoding to use when interpreting percent escapes.

Return Value

A new `CFString` object, or `NULL` if the percent escapes cannot be converted to characters, assuming the encoding given by *encoding*. If no characters need to be replaced, this function returns the original string with its reference count incremented. Ownership follows the Create Rule.

Availability

Available in Mac OS X v10.3 and later.

Declared In

`CFURL.h`

CFURLCreateWithBytes

Creates a `CFURL` object using a given character bytes.

```
CFURLRef CFURLCreateWithBytes (
   CFAllocatorRef allocator,
   const UInt8 *URLBytes,
   CFIndex length,
   CFStringEncoding encoding,
   CFURLRef baseURL
);
```

Parameters

allocator

The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

URLBytes

The character bytes to convert into a `CFURL` object.

length

The number of bytes in *URLBytes*.

encoding

The string encoding of the `URLBytes` string. This encoding is also used to interpret percent escape sequences.

baseURL

The URL to which *URLBytes* is relative. Pass `NULL` if *URLBytes* contains an absolute URL or if you want to create a relative URL. If *URLBytes* contains an absolute URL, this parameter is ignored.

Return Value

A new `CFURL` object. Ownership follows the Create Rule.

Discussion

The specified string encoding will be used both to interpret *URLBytes*, and to interpret any percent-escapes within the string.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

CarbonCocoa_PictureCursor

DisplayURL

ImageBrowserView

RecentItems

Declared In

`CFURL.h`

CFURLCreateWithFileSystemPath

Creates a `CFURL` object using a local file system path string.

```
CFURLRef CFURLCreateWithFileSystemPath (
   CFAllocatorRef allocator,
   CFStringRef filePath,
   CFURLPathStyle pathStyle,
   Boolean isDirectory
);
```

Parameters

allocator

The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

filePath

The path string to convert to a `CFURL` object.

pathStyle

The operating system path style used in `filePath`. See Path Style (page 39) for a list of possible values.

isDirectory

A Boolean value that specifies whether `filePath` is treated as a directory path when resolving against relative path components. Pass `true` if the pathname indicates a directory, `false` otherwise.

Return Value
A new `CFURL` object. Ownership follows the Create Rule.

Discussion
If *`filePath`* is not absolute, the resulting URL will be considered relative to the current working directory (evaluated when this function is being invoked).

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Related Sample Code
ImageBrowserView
QISA
QTExtractAndConvertToMovieFile
Quartz EB
TexturePerformanceDemo

Declared In
`CFURL.h`

CFURLCreateWithFileSystemPathRelativeToBase

Creates a `CFURL` object using a local file system path string relative to a base URL.

```
CFURLRef CFURLCreateWithFileSystemPathRelativeToBase (
   CFAllocatorRef allocator,
   CFStringRef filePath,
   CFURLPathStyle pathStyle,
   Boolean isDirectory,
   CFURLRef baseURL
);
```

Parameters

`allocator`
The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

`filePath`
The path string to convert to a `CFURL` object.

`pathStyle`
The operating system path style used in the *`filePath`* string. See Path Style (page 39) for a list of possible values.

`isDirectory`
A Boolean value that specifies whether *`filePath`* is treated as a directory path when resolving against relative path components. Pass `true` if the pathname indicates a directory, `false` otherwise.

`baseURL`
The base URL against which to resolve the *`filePath`*.

Return Value
A new `CFURL` object. Ownership follows the Create Rule.

Discussion
This function takes a path name in the form of a `CFString` object, resolves it against a base URL, and returns a new `CFURL` object containing the result.

Availability
Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
Aperture Image Resizer

Declared In
`CFURL.h`

CFURLCreateWithString

Creates a `CFURL` object using a given `CFString` object.

```
CFURLRef CFURLCreateWithString (
   CFAllocatorRef allocator,
   CFStringRef URLString,
   CFURLRef baseURL
);
```

Parameters

`allocator`

The allocator to use to allocate memory for the new `CFURL` object. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

`URLString`

The `CFString` object containing the URL string.

`baseURL`

The URL to which `URLString` is relative. Pass `NULL` if *`URLString`* contains an absolute URL or if you want to create a relative URL. If `URLString` contains an absolute URL, `baseURL` is ignored.

Return Value
A new `CFURL` object. Ownership follows the Create Rule.

Discussion
Any escape sequences in *`URLString`* will be interpreted using UTF-8.

Availability
Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
AuthForAll

CFFTPSample

ComboBoxPrefs

DockBrowser

LocalServer

Declared In
`CFURL.h`

CFURLGetBaseURL

Returns the base URL of a given URL if it exists.

```
CFURLRef CFURLGetBaseURL (
   CFURLRef anURL
);
```

Parameters

anURL

The `CFURL` object to examine.

Return Value

A `CFURL` object representing the base URL of *anURL*. Ownership follows the Get Rule.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Declared In

`CFURL.h`

CFURLGetByteRangeForComponent

Returns the range of the specified component in the bytes of a URL.

```
CFRange CFURLGetByteRangeForComponent (
   CFURLRef url,
   CFURLComponentType component,
   CFRange *rangeIncludingSeparators
);
```

Parameters

anURL

The URL containing *component*.

component

The type of component in *anURL* whose range you want to obtain. See Component Type (page 37) for possible values.

rangeIncludingSeparators

Specifies the range of *component* including the sequences that separate component from the previous and next components. If there is no previous or next components, this function will match the range of the component itself. If *anURL* does not contain *component*, *rangeIncludingSeparators* is set to the location where the component would be inserted.

Return Value

The range of bytes for *component* in the buffer returned by the `CFURLGetBytes` (page 33) function. If *anURL* does not contain *component*, the first part of the returned range is set to `kCFNotFound`.

Discussion

This function is intended to be used in conjunction with the `CFURLGetBytes` (page 33) function, since the range returned is only applicable to the bytes returned by `CFURLGetBytes` (page 33).

Availability

Available in Mac OS X v10.3 and later.

Related Sample Code
DisplayURL

Declared In
`CFURL.h`

CFURLGetBytes

Returns by reference the byte representation of a URL object.

```
CFIndex CFURLGetBytes (
   CFURLRef url,
   UInt8 *buffer,
   CFIndex bufferLength
);
```

Parameters

anURL

The URL object to convert to a byte representation.

buffer

The buffer where you want the bytes to be placed. If the buffer is of insufficient size, returns `-1` and no bytes are placed in buffer. If `NULL` the needed length is computed and returned. The returned bytes are the original bytes from which the URL was created. If the URL was created from a string, the bytes are the bytes of the string encoded via UTF-8.

bufferLength

The number of bytes in *buffer*.

Return Value
Returns the number of bytes in *buffer* that were filled. If the buffer is of insufficient size, returns `-1`.

Availability
Available in Mac OS X v10.3 and later.

Related Sample Code
DisplayURL

Declared In
`CFURL.h`

CFURLGetFileSystemRepresentation

Fills a buffer with the file system's native string representation of a given URL's path.

```
Boolean CFURLGetFileSystemRepresentation (
   CFURLRef url,
   Boolean resolveAgainstBase,
   UInt8 *buffer,
   CFIndex maxBufLen
);
```

Parameters

url

The `CFURL` object whose native file system representation you want to obtain.

resolveAgainstBase

Pass `true` to return an absolute path name.

buffer

A pointer to a character buffer. On return, the buffer holds the native file system's representation of *url*. The buffer is null-terminated. This parameter must be at least *maxBufLen* in size for the file system in question to avoid failures for insufficiently large buffers.

maxBufLen

The maximum number of characters that can be written to *buffer*.

Return Value
`true` if successful, `false` if an error occurred.

Discussion
No more than *maxBufLen* bytes are written to *buffer*. If *url* requires more than *maxBufLen* bytes to represent itself, including the terminating null byte, this function returns `false`. To avoid this possible failure, you should pass a buffer with size of at least the maximum path length for the file system in question.

Availability
Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
BSDLLCTest

CheckExecutableArchitecture

MemoryBasedBundle

MoreIsBetter

QISA

Declared In
`CFURL.h`

CFURLGetFSRef

Converts a given URL to a file or directory object.

```
Boolean CFURLGetFSRef (
   CFURLRef url,
   struct FSRef *fsRef
);
```

Parameters

url

The `CFURL` object to convert to a file or directory object.

fsRef

Upon return, contains the file or directory object representing *url*.

Return Value
`true` if the conversion was successful, otherwise `false`.

Special Considerations

The function cannot create an `FSRef` object if the path specified by `url` contains an alias. The function can, however, traverse symbolic links.

Availability
Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Declared In
`CFURL.h`

CFURLGetPortNumber

Returns the port number from a given URL.

```
SInt32 CFURLGetPortNumber (
   CFURLRef anURL
);
```

Parameters

`anURL`
The `CFURL` object to examine.

Return Value
The port number of *`anURL`*, or `-1` if no port number exists.

Availability
Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
ImageClient

Declared In
`CFURL.h`

CFURLGetString

Returns the URL as a `CFString` object.

```
CFStringRef CFURLGetString (
   CFURLRef anURL
);
```

Parameters

`anURL`
The `CFURL` object to convert into a `CFString` object.

Return Value
A string representation of *`anURL`*. Ownership follows the Get Rule.

Availability
Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
AlbumToSlideshow

LoginItemsAE

QISA

Declared In
`CFURL.h`

CFURLGetTypeID

Returns the type identifier for the `CFURL` opaque type.

```
CFTypeID CFURLGetTypeID (
   void
);
```

Return Value
The type identifier for the `CFURL` opaque type.

Availability
Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
LoginItemsAE

Declared In
`CFURL.h`

CFURLHasDirectoryPath

Determines if a given URL's path represents a directory.

```
Boolean CFURLHasDirectoryPath (
   CFURLRef anURL
);
```

Parameters

`anURL`

The `CFURL` object to examine.

Return Value
`true` if *`anURL`* represents a directory, `false` otherwise.

Availability
Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code
CFFTPSample

ImageClient

MoreAppleEvents

MoreIsBetter

QISA

Declared In
`CFURL.h`

Data Types

CFURLRef

A reference to a CFURL object.

```
typedef const struct __CFURL *CFURLRef;
```

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

Declared In
`CFURL.h`

Constants

Component Type

The types of components in a URL.

```
typedef enum {
    kCFURLComponentScheme = 1,
    kCFURLComponentNetLocation = 2,
    kCFURLComponentPath = 3,
    kCFURLComponentResourceSpecifier = 4,
    kCFURLComponentUser = 5,
    kCFURLComponentPassword = 6,
    kCFURLComponentUserInfo = 7,
    kCFURLComponentHost = 8,
    kCFURLComponentPort = 9,
    kCFURLComponentParameterString = 10,
    kCFURLComponentQuery = 11,
    kCFURLComponentFragment = 12
} CFURLComponentType;
typedef enum CFURLPathStyle CFURLPathStyle;
```

Constants

`kCFURLComponentScheme`

The URL's scheme.

Available in Mac OS X v10.3 and later.

Declared in `CFURL.h`.

`kCFURLComponentNetLocation`

The URL's network location.

Available in Mac OS X v10.3 and later.

Declared in `CFURL.h`.

`kCFURLComponentPath`
: The URL's path component.

  Available in Mac OS X v10.3 and later.

  Declared in `CFURL.h`.

`kCFURLComponentResourceSpecifier`
: The URL's resource specifier.

  Available in Mac OS X v10.3 and later.

  Declared in `CFURL.h`.

`kCFURLComponentUser`
: The URL's user.

  Available in Mac OS X v10.3 and later.

  Declared in `CFURL.h`.

`kCFURLComponentPassword`
: The user's password.

  Available in Mac OS X v10.3 and later.

  Declared in `CFURL.h`.

`kCFURLComponentUserInfo`
: The user's information.

  Available in Mac OS X v10.3 and later.

  Declared in `CFURL.h`.

`kCFURLComponentHost`
: The URL's host.

  Available in Mac OS X v10.3 and later.

  Declared in `CFURL.h`.

`kCFURLComponentPort`
: The URL's port.

  Available in Mac OS X v10.3 and later.

  Declared in `CFURL.h`.

`kCFURLComponentParameterString`
: The URL's parameter string.

  Available in Mac OS X v10.3 and later.

  Declared in `CFURL.h`.

`kCFURLComponentQuery`
: The URL's query.

  Available in Mac OS X v10.3 and later.

  Declared in `CFURL.h`.

`kCFURLComponentFragment`
: The URL's fragment.

  Available in Mac OS X v10.3 and later.

  Declared in `CFURL.h`.

Discussion

These constants are used by the `CFURLGetByteRangeForComponent` (page 32) function.

Availability
Available in Mac OS X v10.3 and later.

Path Style

Options you can use to determine how CFURL functions parse a file system path name.

```
enum CFURLPathStyle {
    kCFURLPOSIXPathStyle = 0,
    kCFURLHFSPathStyle = 1,
    kCFURLWindowsPathStyle = 2
};
typedef enum CFURLPathStyle CFURLPathStyle;
```

Constants

`kCFURLPOSIXPathStyle`

Indicates a POSIX style path name. Components are slash delimited. A leading slash indicates an absolute path; a trailing slash is not significant.

Available in Mac OS X v10.0 and later.

Declared in `CFURL.h`.

`kCFURLHFSPathStyle`

Indicates a HFS style path name. Components are colon delimited. A leading colon indicates a relative path, otherwise the first path component denotes the volume.

Available in Mac OS X v10.0 and later.

Declared in `CFURL.h`.

`kCFURLWindowsPathStyle`

Indicates a Windows style path name.

Available in Mac OS X v10.0 and later.

Declared in `CFURL.h`.

Document Revision History

This table describes the changes to *CFURL Reference*.

Date	Notes
2009-02-04	Corrected typos.
2008-07-11	Clarified the description of the CFURLGetBytes function.
2006-01-10	Clarified the behavior of the functions CFURLCreateStringByAddingPercentEscapes and CFURLGetFSRef.
2005-12-06	Made minor changes to clarify memory management rules.
2005-11-09	Removed reference to retired document.
2005-10-04	Corrected minor typographic errors.
2005-07-07	Clarified implementations of CFURLCanBeDecomposed and CFURLCreateCopyAppendingPathComponent, and description of CFURLCopyPathExtension.
2005-04-29	Moved Introduction to new Introduction page.
2004-08-31	Clarification of return values for CFURLCopyLastPathComponent.
2003-08-01	Added descriptions of new Mac OS X v10.3 API.
2003-01-01	First version of this document.

Index