
NSBundle Class Reference

Data Management: File Management



2009-07-23



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

iPhone is a trademark of Apple Inc.

Intel and Intel Core are registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

PowerPC and the PowerPC logo are trademarks of International Business Machines Corporation, used under license therefrom.

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

NSBundle Class Reference 5

Overview	5
Tasks	6
Initializing an NSBundle	6
Getting an NSBundle	6
Getting a Bundled Class	6
Finding Resources	6
Getting the Bundle Directory	7
Getting Bundle Information	8
Managing Localized Resources	8
Loading a Bundle's Code	9
Managing Localizations	9
Class Methods	9
allBundles	9
allFrameworks	10
bundleForClass:	10
bundleWithIdentifier:	11
bundleWithPath:	11
bundleWithURL:	12
mainBundle	13
pathForResource ofType:inDirectory:	13
pathsForResourceOfType:inDirectory:	14
preferredLocalizationsFromArray:	15
preferredLocalizationsFromArray:forPreferences:	16
URLForResource:withExtension:subdirectory:inBundleWithURL:	16
URLsForResourceWithExtension:subdirectory:inBundleWithURL:	17
Instance Methods	18
builtinPlugInsPath	18
builtinPlugInsURL	18
bundleIdentifier	18
bundlePath	19
bundleURL	19
classNameNamed:	20
developmentLocalization	20
executableArchitectures	21
executablePath	21
executableURL	21
infoDictionary	22
initWithPath:	22
initWithURL:	23
isLoading	24

- load 24
- loadAndReturnError: 25
- localizations 26
- localizedInfoDictionary 26
- localizedStringForKey:value:table: 26
- objectForInfoDictionaryKey: 28
- pathForAuxiliaryExecutable: 28
- pathForResource:ofType: 29
- pathForResource:ofType:inDirectory: 30
- pathForResource:ofType:inDirectory:forLocalization: 30
- pathsForResourcesOfType:inDirectory: 31
- pathsForResourcesOfType:inDirectory:forLocalization: 32
- preferredLocalizations 33
- preflightAndReturnError: 33
- principalClass 34
- privateFrameworksPath 35
- privateFrameworksURL 35
- resourcePath 36
- resourceURL 36
- sharedFrameworksPath 37
- sharedFrameworksURL 37
- sharedSupportPath 37
- sharedSupportURL 38
- unload 38
- URLForAuxiliaryExecutable: 39
- URLForResource:withExtension: 39
- URLForResource:withExtension:subdirectory: 40
- URLForResource:withExtension:subdirectory:localization: 41
- URLsForResourcesWithExtension:subdirectory: 41
- URLsForResourcesWithExtension:subdirectory:localization: 42
- Constants 43
 - Mach-O Architecture 43
 - NSLoadedClasses 43
- Notifications 44
 - NSBundleDidLoadNotification 44

Document Revision History 45

Index 47

NSBundle Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/Foundation.framework
Availability	Available in Mac OS X v10.0 and later.
Declared in	NSBundle.h
Companion guides	Bundle Programming Guide Resource Programming Guide
Related sample code	CoreRecipes GLSLShowpiece NumberInput_IMKit_Sample OutputBins2PDE Quartz Composer WWDC 2005 TextEdit

Overview

An `NSBundle` object represents a location in the file system that groups code and resources that can be used in a program. `NSBundle` objects locate program resources, dynamically load and unload executable code, and assist in localization. You build a bundle in Xcode using one of these project types: Application, Framework, plug-ins.

Although bundle structures vary depending on the target platform and the type of bundle you are building, the `NSBundle` class hides this underlying structure in most (but not all) cases. Many of the methods you use to load resources from a bundle automatically locate the appropriate starting directory and look for resources in known places. For information about application bundle structures (for Mac OS X and iPhone OS), see *Bundle Programming Guide*. For information about the structure of framework bundles, see *Framework Programming Guide*. For information about the structure of Mac OS X plug-ins, see *Code Loading Programming Topics for Cocoa*.

For additional information about how to load nib files and images in a Mac OS X application, see *NSBundle Additions Reference*. For information about how to load nib files in an iPhone application, see *NSBundle UIKit Additions Reference*.

Unlike some other Foundation classes with corresponding Core Foundation names (such as `NSString` and `CFString`), `NSBundle` objects cannot be cast (“toll-free bridged”) to `CFBundle` references. If you need functionality provided in `CFBundle`, you can still create a `CFBundle` and use the `CFBundle` API. See *Interchangeable Data Types* for more information on toll-free bridging.

Tasks

Initializing anNSBundle

- + [bundleWithURL:](#) (page 12)
Returns an `NSBundle` object that corresponds to the specified file URL.
- + [bundleWithPath:](#) (page 11)
Returns an `NSBundle` object that corresponds to the specified directory.
- [initWithURL:](#) (page 23)
Returns an `NSBundle` object initialized to correspond to the specified file URL.
- [initWithPath:](#) (page 22)
Returns an `NSBundle` object initialized to correspond to the specified directory.

Getting anNSBundle

- + [bundleForClass:](#) (page 10)
Returns the `NSBundle` object with which the specified class is associated.
- + [bundleWithIdentifier:](#) (page 11)
Returns the previously created `NSBundle` instance that has the specified bundle identifier.
- + [mainBundle](#) (page 13)
Returns the `NSBundle` object that corresponds to the directory where the current application executable is located.
- + [allBundles](#) (page 9)
Returns an array of all the application's non-framework bundles.
- + [allFrameworks](#) (page 10)
Returns an array of all of the application's bundles that represent frameworks.

Getting a Bundled Class

- [classNameNamed:](#) (page 20)
Returns the `Class` object for the specified name.
- [principalClass](#) (page 34)
Returns the receiver's principal class.

Finding Resources

- [URLForResource:withExtension:subdirectory:](#) (page 40)
Returns the file URL for the resource file identified by the specified name and extension and residing in a given bundle directory.
- + [pathForResource ofType:inDirectory:](#) (page 13)
Returns the full pathname for the resource file identified by the specified name and extension and residing in a given bundle directory.

- [URLForResource:withExtension:](#) (page 39)
Returns the file URL for the resource identified by the specified name and file extension.
- [pathForResource ofType:](#) (page 29)
Returns the full pathname for the resource identified by the specified name and file extension.
- [URLsForResourceWithExtension:subdirectory:](#) (page 41)
Returns the file URL for the resource identified by the specified name and file extension and located in the specified bundle subdirectory.
- [pathForResource ofType:inDirectory:](#) (page 30)
Returns the full pathname for the resource identified by the specified name and file extension and located in the specified bundle subdirectory.
- [URLForResource:withExtension:subdirectory:localization:](#) (page 41)
Returns the file URL for the resource identified by the specified name and file extension, located in the specified bundle subdirectory, and limited to global resources and those associated with the specified localization.
- [pathForResource ofType:inDirectory:forLocalization:](#) (page 30)
Returns the full pathname for the resource identified by the specified name and file extension, located in the specified bundle subdirectory, and limited to global resources and those associated with the specified localization.
- + [pathsForResourceOfType:inDirectory:](#) (page 14)
Returns an array containing the pathnames for all bundle resources having the specified extension and residing in the bundle directory at the specified path.
- [pathsForResourceOfType:inDirectory:](#) (page 31)
Returns an array containing the pathnames for all bundle resources having the specified filename extension and residing in the resource subdirectory.
- [URLsForResourceWithExtension:subdirectory:localization:](#) (page 42)
Returns an array containing the file URLs for all bundle resources having the specified filename extension, residing in the specified resource subdirectory, and limited to global resources and those associated with the specified localization.
- [pathsForResourceOfType:inDirectory:forLocalization:](#) (page 32)
Returns an array containing the file URLs for all bundle resources having the specified filename extension, residing in the specified resource subdirectory, and limited to global resources and those associated with the specified localization.
- + [URLForResource:withExtension:subdirectory:inBundleWithURL:](#) (page 16)
Creates and returns a file URL for the resource with the specified name and extension in the specified bundle.
- + [URLsForResourceWithExtension:subdirectory:inBundleWithURL:](#) (page 17)
Returns an array containing the file URLs for all bundle resources having the specified filename extension, residing in the specified resource subdirectory, within the specified bundle.
- [resourcePath](#) (page 36)
Returns the full pathname of the receiving bundle's subdirectory containing resources.

Getting the Bundle Directory

- [bundleURL](#) (page 19)
Returns the full URL of the receiver's bundle directory.

- [bundlePath](#) (page 19)
Returns the full pathname of the receiver's bundle directory.

Getting Bundle Information

- [bundleIdentifier](#) (page 18)
Returns the receiver's bundle identifier.
- [infoDictionary](#) (page 22)
Returns a dictionary that contains information about the receiver.
- [objectForInfoDictionaryKey:](#) (page 28)
Returns the value associated with the specified key in the receiver's information property list.
- [builtInPlugInsURL](#) (page 18)
Returns the file URL of the receiver's subdirectory containing plug-ins.
- [builtInPlugInsPath](#) (page 18)
Returns the full pathname of the receiver's subdirectory containing plug-ins.
- [executableURL](#) (page 21)
Returns the file URL of the receiver's executable file.
- [executablePath](#) (page 21)
Returns the full pathname of the receiver's executable file.
- [URLForAuxiliaryExecutable:](#) (page 39)
Returns the file URL of the executable with the specified name in the receiver's bundle.
- [pathForAuxiliaryExecutable:](#) (page 28)
Returns the full pathname of the executable with the specified name in the receiver's bundle.
- [privateFrameworksURL](#) (page 35)
Returns the file URL of the receiver's subdirectory containing private frameworks.
- [privateFrameworksPath](#) (page 35)
Returns the full pathname of the receiver's subdirectory containing private frameworks.
- [sharedFrameworksURL](#) (page 37)
Returns the file URL of the receiver's subdirectory containing shared frameworks.
- [sharedFrameworksPath](#) (page 37)
Returns the full pathname of the receiver's subdirectory containing shared frameworks.
- [sharedSupportURL](#) (page 38)
Returns the file URL of the receiver's subdirectory containing shared support files.
- [sharedSupportPath](#) (page 37)
Returns the full pathname of the receiver's subdirectory containing shared support files.
- [resourceURL](#) (page 36)
Returns the file URL of the receiver's subdirectory containing resource files.

Managing Localized Resources

- [localizedStringForKey:value:table:](#) (page 26)
Returns a localized version of the string designated by the specified key and residing in the specified table.

Loading a Bundle's Code

- [executableArchitectures](#) (page 21)
Returns an array of numbers indicating the architecture types supported by the bundle's executable.
- [preflightAndReturnError:](#) (page 33)
Returns a Boolean value indicating whether the bundle's executable code could be loaded successfully.
- [load](#) (page 24)
Dynamically loads the bundle's executable code into a running program, if the code has not already been loaded.
- [loadAndReturnError:](#) (page 25)
Loads the bundle's executable code and returns any errors.
- [isLoading](#) (page 24)
Obtains information about the load status of a bundle.
- [unload](#) (page 38)
Unloads the code associated with the receiver.

Managing Localizations

- + [preferredLocalizationsFromArray:](#) (page 15)
Returns one or more localizations from the specified list that a bundle object would use to locate resources for the current user.
- + [preferredLocalizationsFromArray:forPreferences:](#) (page 16)
Returns the localizations that a bundle object would prefer, given the specified bundle and user preference localizations.
- [localizations](#) (page 26)
Returns a list of all the localizations contained within the receiver's bundle.
- [developmentLocalization](#) (page 20)
Returns the localization used to create the bundle.
- [preferredLocalizations](#) (page 33)
Returns an array of strings indicating the actual localizations contained in the receiver's bundle.
- [localizedInfoDictionary](#) (page 26)
Returns a dictionary with the keys from the bundle's localized property list.

Class Methods

allBundles

Returns an array of all the application's non-framework bundles.

```
+ (NSArray *)allBundles
```

Return Value

An array of all the application's non-framework bundles.

Discussion

The returned array includes the main bundle and all bundles that have been dynamically created but doesn't contain any bundles that represent frameworks.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSBundle.h

allFrameworks

Returns an array of all of the application's bundles that represent frameworks.

```
+ (NSArray *)allFrameworks
```

Return Value

An array of all of the application's bundles that represent frameworks. Only frameworks with one or more Objective-C classes in them are included.

Discussion

The returned array includes frameworks that are linked into an application when the application is built and bundles for frameworks that have been dynamically created.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

Core Data HTML Store

CoreRecipes

Declared In

NSBundle.h

bundleForClass:

Returns the `NSBundle` object with which the specified class is associated.

```
+ (NSBundle *)bundleForClass:(Class)aClass
```

Parameters

aClass

A class.

Return Value

The `NSBundle` object that dynamically loaded *aClass* (a loadable bundle), the `NSBundle` object for the framework in which *aClass* is defined, or the main bundle object if *aClass* was not dynamically loaded or is not defined in a framework.

Availability

Available in Mac OS X v10.0 and later.

See Also

+ [mainBundle](#) (page 13)

+ [bundleWithPath:](#) (page 11)

Related Sample Code

BundleLoader

ComplexBrowser

Core Data HTML Store

CoreRecipes

GLSLShowpiece

Declared In

NSBundle.h

bundleWithIdentifier:

Returns the previously created `NSBundle` instance that has the specified bundle identifier.

```
+ (NSBundle *)bundleWithIdentifier:(NSString *)identifier
```

Parameters

identifier

The identifier for an existing `NSBundle` instance.

Return Value

The previously created `NSBundle` instance that has the bundle identifier *identifier*. Returns `nil` if the requested bundle is not found.

Discussion

This method is typically used by frameworks and plug-ins to locate their own bundle at runtime. This method may be somewhat more efficient than trying to locate the bundle using the [bundleForClass:](#) (page 10) method.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CIRAWFilterSample

ImageApp

PrefsPane

Declared In

NSBundle.h

bundleWithPath:

Returns an `NSBundle` object that corresponds to the specified directory.

```
+ (NSBundle *)bundleWithPath:(NSString *)fullPath
```

Parameters

fullPath

The path to a directory. This must be a full pathname for a directory; if it contains any symbolic links, they must be resolvable.

Return Value

The `NSBundle` object that corresponds to *fullPath*, or `nil` if *fullPath* does not identify an accessible bundle directory.

Discussion

This method allocates and initializes the returned object if there is no existing `NSBundle` associated with *fullPath*, in which case it returns the existing object.

Availability

Available in Mac OS X v10.0 and later.

See Also

- + [mainBundle](#) (page 13)
- + [bundleForClass:](#) (page 10)
- [initWithPath:](#) (page 22)
- + [bundleWithURL:](#) (page 12)

Related Sample Code

BundleLoader
CocoaAUHost
Core Data HTML Store
QTAudioContextInsert
SimpleScriptingPlugin

Declared In

`NSBundle.h`

bundleWithURL:

Returns an `NSBundle` object that corresponds to the specified file URL.

```
+ (NSBundle *)bundleWithURL:(NSURL *)url
```

Parameters

url

The URL to a directory. This must be a URL for a directory; if it contains any symbolic links, they must be resolvable.

Return Value

The `NSBundle` object that corresponds to *url*, or `nil` if *url* does not identify an accessible bundle directory.

Discussion

This method allocates and initializes the returned object if there is no existing `NSBundle` associated with *url*, in which case it returns the existing object.

Availability

Available in Mac OS X v10.6 and later.

See Also

- + [bundleWithPath:](#) (page 11)
- + [bundleWithIdentifier:](#) (page 11)
- + [bundleForClass:](#) (page 10)

Declared In

NSBundle.h

mainBundle

Returns the `NSBundle` object that corresponds to the directory where the current application executable is located.

```
+ (NSBundle *)mainBundle
```

Return Value

The `NSBundle` object that corresponds to the directory where the application executable is located, or `nil` if a bundle object could not be created.

Discussion

This method allocates and initializes a bundle object if one doesn't already exist. The new object corresponds to the directory where the application executable is located. Be sure to check the return value to make sure you have a valid bundle. This method may return a valid bundle object even for unbundled applications.

In general, the main bundle corresponds to an application file package or application wrapper: a directory that bears the name of the application and is marked by a ".app" extension.

Availability

Available in Mac OS X v10.0 and later.

See Also

+ [bundleForClass:](#) (page 10)

+ [bundleWithPath:](#) (page 11)

Related Sample Code

CITransitionSelectorSample2

CoreRecipes

FunHouse

ImageKitDemo

NumberInput_IMKit_Sample

Declared In

NSBundle.h

pathForResource ofType:inDirectory:

Returns the full pathname for the resource file identified by the specified name and extension and residing in a given bundle directory.

```
+ (NSString *)pathForResource:(NSString *)name ofType:(NSString *)extension
  inDirectory:(NSString *)bundlePath
```

Parameters

name

The name of a resource file contained in the directory specified by *bundlePath*.

extension

If *extension* is an empty string or `nil`, the extension is assumed not to exist and the file URL is the first file encountered that exactly matches *name*.

bundlePath

The path of a top-level bundle directory. This must be a valid path. For example, to specify the bundle directory for a Mac OS X application, you might specify the path `/Applications/MyApp.app`.

Return Value

The full pathname for the resource file or `nil` if the file could not be located. This method also returns `nil` if the bundle specified by the `bundlePath` parameter does not exist or is not a readable directory.

Discussion

The method first looks for a matching resource file in the non-localized resource directory of the specified bundle. (In Mac OS X, this directory is typically called `Resources` but in iPhone OS, it is the main bundle directory.) If a matching resource file is not found, it then looks in the top level of any available language-specific “.lproj” directories. (The search order for the language-specific directories corresponds to the user’s preferences.) It does not recurse through other subdirectories at any of these locations. For more details see Bundles and Localization.

Note: This method is best suited only for the occasional retrieval of resource files. In most cases where you need to retrieve bundle resources, it is preferable to use the `NSBundle` instance methods instead.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [localizedStringForKey:value:table:](#) (page 26)
- [pathForResource ofType:](#) (page 29)
- [pathForResource ofType:inDirectory:](#) (page 30)
- + [pathsForResourcesOfType:inDirectory:](#) (page 14)
- [pathsForResourcesOfType:inDirectory:](#) (page 31)
- [URLForResource:withExtension:subdirectory:](#) (page 40)

Related Sample Code

OpenALExample

Declared In

`NSBundle.h`

pathsForResourcesOfType:inDirectory:

Returns an array containing the pathnames for all bundle resources having the specified extension and residing in the bundle directory at the specified path.

```
+ (NSArray *)pathsForResourcesOfType:(NSString *)extension inDirectory:(NSString *)bundlePath
```

Parameters

extension

If *extension* is an empty string or `nil`, the extension is assumed not to exist and the file URL is the first file encountered that exactly matches *name*.

bundlePath

The top-level directory of a bundle. This must represent a valid path.

Return Value

An array containing the full pathnames for all bundle resources with the specified extension. This method returns an empty array if no matching resource files are found. It also returns an empty array if the bundle specified by the `bundlePath` parameter does not exist or is not a readable directory.

Discussion

This method provides a means for dynamically discovering multiple bundle resources of the same type.

The method first looks for matching resource files in the nonlocalized resource directory of the specified bundle. (In Mac OS X, this directory is typically called `Resources` but in iPhone OS, it is the main bundle directory.) It then looks in the top level of any available language-specific “.lproj” directories. It does not recurse through other subdirectories at any of these locations. For more details see Bundles and Localization.

Note: This method is best suited only for the occasional retrieval of resource files. In most cases where you need to retrieve bundle resources, it is preferable to use the `NSBundle` instance methods instead.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [localizedStringForKey:value:table:](#) (page 26)
- [pathForResource ofType:](#) (page 29)
- [pathForResource ofType: inDirectory:](#) (page 30)
- + [pathForResource ofType: inDirectory:](#) (page 13)

Related Sample Code

AutoSample
CocoaCreateMovie
OpenALExample
SimpleScriptingPlugin

Declared In

`NSBundle.h`

preferredLocalizationsFromArray:

Returns one or more localizations from the specified list that a bundle object would use to locate resources for the current user.

```
+ (NSArray *)preferredLocalizationsFromArray:(NSArray *)localizationsArray
```

Parameters

localizationsArray

An array of `NSString` objects, each of which specifies the name of a localization that the bundle supports.

Return Value

An array of `NSString` objects containing the preferred localizations. These strings are ordered in the array according to the current user's language preferences and are taken from the strings in the `localizationsArray` parameter.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSBundle.h`

preferredLocalizationsFromArray:forPreferences:

Returns the localizations that a bundle object would prefer, given the specified bundle and user preference localizations.

```
+ (NSArray *)preferredLocalizationsFromArray:(NSArray *)localizationsArray
    forPreferences:(NSArray *)preferencesArray
```

Parameters

localizationsArray

An array of `NSString` objects, each of which specifies the name of a localization that the bundle supports.

preferencesArray

An array of `NSString` objects containing the user's preferred localizations. If this parameter is `nil`, the method uses the current user's localization preferences.

Return Value

An array of `NSString` objects containing the preferred localizations. These strings are ordered in the array according to the specified preferences and are taken from the strings in the `localizationsArray` parameter.

Discussion

Use the argument `localizationsArray` to specify the supported localizations of the bundle and use `preferencesArray` to specify the user's localization preferences.

If none of the user-preferred localizations are available in the bundle, this method chooses one of the bundle localizations and returns it.

Availability

Available in Mac OS X v10.2 and later.

Declared In

`NSBundle.h`

URLForResource:withExtension:subdirectory:inBundleWithURL:

Creates and returns a file URL for the resource with the specified name and extension in the specified bundle.

```
+ (NSURL *)URLForResource:(NSString *)name withExtension:(NSString *)ext
    subdirectory:(NSString *)subpath inBundleWithURL:(NSURL *)bundleURL
```

Parameters*name*

The name of the resource file.

*extension*If *extension* is an empty string or `nil`, the extension is assumed not to exist and the file URL is the first file encountered that exactly matches *name*.*subpath*

The name of the bundle subdirectory to search.

bundleURL

The file URL of the bundle to search.

Return ValueThe file URL for the resource file or `nil` if the file could not be located.**Availability**

Available in Mac OS X v10.6 and later.

Declared In

NSBundle.h

URLsForResourcesWithExtension:subdirectory:inBundleWithURL:

Returns an array containing the file URLs for all bundle resources having the specified filename extension, residing in the specified resource subdirectory, within the specified bundle.

```
+ (NSArray *)URLsForResourcesWithExtension:(NSString *)ext subdirectory:(NSString *)subpath inBundleWithURL:(NSURL *)bundleURL
```

Parameters*ext*

The file extension of the files to retrieve.

subpath

The name of the bundle subdirectory to search.

bundleURL

The file URL of the bundle to search.

Return ValueThe file URL for the resource file or `nil` if the file could not be located.**Availability**

Available in Mac OS X v10.6 and later.

Declared In

NSBundle.h

Instance Methods

builtInPlugInsPath

Returns the full pathname of the receiver's subdirectory containing plug-ins.

```
- (NSString *)builtInPlugInsPath
```

Return Value

The full pathname of the receiving bundle's subdirectory containing plug-ins.

Discussion

This method returns the appropriate path for modern application and framework bundles. This method may not return a path for non-standard bundle formats or for some older bundle formats.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

BundleLoader

CIAnnotation

CIColorTracking

Core Data HTML Store

SimpleScriptingPlugin

Declared In

NSBundle.h

builtInPlugInsURL

Returns the file URL of the receiver's subdirectory containing plug-ins.

```
- (NSURL *)builtInPlugInsURL
```

Return Value

The file URL of the receiving bundle's subdirectory containing plug-ins.

Discussion

This method returns the appropriate path for modern application and framework bundles. This method may not return a URL for non-standard bundle formats or for some older bundle formats.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSBundle.h

bundleIdentifier

Returns the receiver's bundle identifier.

- (NSString *)bundleIdentifier

Return Value

The receiver's bundle identifier, which is defined by the `CFBundleIdentifier` key in the bundle's information property list.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [infoDictionary](#) (page 22)

Related Sample Code

BetterAuthorizationSample

CoreRecipes

NumberInput_IMKit_Sample

StickiesWithCoreData

Declared In

NSBundle.h

bundlePath

Returns the full pathname of the receiver's bundle directory.

- (NSString *)bundlePath

Return Value

The full pathname of the receiver's bundle directory.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

GLUT

OutputBinsPDE

SimpleScriptingPlugin

Declared In

NSBundle.h

bundleURL

Returns the full URL of the receiver's bundle directory.

- (NSURL *)bundleURL

Return Value

The full URL of the receiver's bundle directory.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSBundle.h

classNamed:Returns the `Class` object for the specified name.

- (Class)classNamed:(NSString *)className

Parameters*className*

The name of a class.

Return ValueThe `Class` object for *className*. Returns `nil` if *className* is not one of the classes associated with the receiver or if there is an error loading the executable code containing the class implementation.**Discussion**

If the bundle's executable code is not yet loaded, this method dynamically loads it into memory. Classes (and categories) are loaded from just one file within the bundle directory; this code file has the same name as the directory, but without the extension (`".bundle"`, `".app"`, `".framework"`). As a side effect of code loading, the receiver posts `NSBundleDidLoadNotification` (page 44) after all classes and categories have been loaded; see [“Notifications”](#) (page 44) for details.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [principalClass](#) (page 34)
- [load](#) (page 24)

Related Sample Code

CocoaAUHost

ImageApp

QTAudioContextInsert

Declared In

NSBundle.h

developmentLocalization

Returns the localization used to create the bundle.

- (NSString *)developmentLocalization

Return Value

The localization used to create the bundle.

Discussion

The returned localization corresponds to the value in the `CFBundleDevelopmentRegion` key of the bundle's property list (`Info.plist`).

Availability

Available in Mac OS X v10.2 and later.

Declared In

NSBundle.h

executableArchitectures

Returns an array of numbers indicating the architecture types supported by the bundle's executable.

- (NSArray *)executableArchitectures

Return Value

An array of `NSNumber` objects, each of which contains an integer value corresponding to a supported processor architecture. For a list of common architecture types, see the constants in [“Mach-O Architecture”](#) (page 43). If the bundle does not contain a Mach-O executable, this method returns `nil`.

Discussion

This method scans the bundle's Mach-O executable and returns all of the architecture types it finds. Because they are taken directly from the executable, the returned values may not always correspond to one of the well-known CPU types defined in [“Mach-O Architecture”](#) (page 43).

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSBundle.h

executablePath

Returns the full pathname of the receiver's executable file.

- (NSString *)executablePath

Return Value

The full pathname of the receiving bundle's executable file.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSBundle.h

executableURL

Returns the file URL of the receiver's executable file.

- (NSURL *)executableURL

Return Value

The file URL of the receiving bundle's executable file.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSBundle.h

infoDictionary

Returns a dictionary that contains information about the receiver.

```
- (NSDictionary *)infoDictionary
```

Return Value

A dictionary, constructed from the bundle's `Info.plist` file, that contains information about the receiver. If the bundle does not contain an `Info.plist` file, a valid dictionary is returned but this dictionary contains only private keys that are used internally by the `NSBundle` class. The `NSBundle` class may add extra keys to this dictionary for its own use.

Discussion

Common keys for accessing the values of the dictionary are `CFBundleIdentifier`, `NSMainNibFile`, and `NSPrincipalClass`.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [principalClass](#) (page 34)
+ `dictionaryWithContentsOfFile:` (`NSDictionary`)

Related Sample Code

GLUT

PrefsPane

SimpleScriptingPlugin

VertexPerformanceTest

Declared In

NSBundle.h

initWithPath:

Returns an `NSBundle` object initialized to correspond to the specified directory.

```
- (id)initWithPath:(NSString *)fullPath
```

Parameters

fullPath

The path to a directory. This must be a full pathname for a directory; if it contains any symbolic links, they must be resolvable.

Return Value

An `NSBundle` object initialized to correspond to `fullPath`. This method initializes and returns a new instance only if there is no existing bundle associated with `fullPath`, otherwise it deallocates `self` and returns the existing object. If `fullPath` doesn't exist or the user doesn't have access to it, returns `nil`.

Discussion

It's not necessary to allocate and initialize an instance for the main bundle; use the `mainBundle` (page 13) class method to get this instance. You can also use the `bundleWithPath:` (page 11) class method to obtain a bundle identified by its directory path.

Availability

Available in Mac OS X v10.0 and later.

See Also

- + `bundleForClass:` (page 10)
- `initWithURL:` (page 23)

Declared In

`NSBundle.h`

initWithURL:

Returns an `NSBundle` object initialized to correspond to the specified file URL.

```
- (id)initWithURL:(NSURL *)url
```

Parameters

`url`

The file URL to a directory. This must be a full URL for a directory; if it contains any symbolic links, they must be resolvable.

Return Value

An `NSBundle` object initialized to correspond to `url`. This method initializes and returns a new instance only if there is no existing bundle associated with `url`, otherwise it deallocates `self` and returns the existing object. If `url` doesn't exist or the user doesn't have access to it, returns `nil`.

Discussion

It's not necessary to allocate and initialize an instance for the main bundle; use the `mainBundle` (page 13) class method to get this instance. You can also use the `bundleWithURL:` (page 12) class method to obtain a bundle identified by its file URL.

Availability

Available in Mac OS X v10.6 and later.

See Also

- + `bundleWithPath:` (page 11)
- + `bundleWithIdentifier:` (page 11)
- + `bundleForClass:` (page 10)
- + `bundleWithURL:` (page 12)

Declared In

`NSBundle.h`

isLoading

Obtains information about the load status of a bundle.

- (BOOL)isLoading

Return Value

YES if the bundle's code is currently loaded, otherwise NO.

Availability

Available in Mac OS X v10.2 and later.

See Also

- [load](#) (page 24)

Declared In

NSBundle.h

load

Dynamically loads the bundle's executable code into a running program, if the code has not already been loaded.

- (BOOL)load

Return Value

YES if the method successfully loads the bundle's code or if the code has already been loaded, otherwise NO.

Discussion

You can use this method to load the code associated with a dynamically loaded bundle, such as a plug-in or framework. Prior to Mac OS X version 10.5, a bundle would attempt to load its code—if it had any—only once. Once loaded, you could not unload that code. In Mac OS X version 10.5 and later, you can unload a bundle's executable code using the [unload](#) (page 38) method.

You don't need to load a bundle's executable code to search the bundle's resources.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [loadAndReturnError:](#) (page 25)
- [isLoading](#) (page 24)
- [unload](#) (page 38)
- [className:](#) (page 20)
- [principalClass](#) (page 34)

Related Sample Code

Core Data HTML Store

Declared In

NSBundle.h

loadAndReturnError:

Loads the bundle's executable code and returns any errors.

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

Parameters

error

On input, a pointer to an error object variable. On output, this variable may contain an error object indicating why the bundle's executable could not be loaded. If no error occurred, this parameter is left unmodified. You may specify `nil` for this parameter if you are not interested in the error information.

Return Value

YES if the bundle's executable code was loaded successfully or was already loaded; otherwise, NO if the code could not be loaded.

Discussion

If this method returns NO and you pass a value for the *error* parameter, a suitable error object is returned in that parameter. Potential errors returned are in the Cocoa error domain and include the types that follow. For a full list of error types, see `FoundationErrors.h`.

- `NSFileNoSuchFileError` - returned if the bundle's executable file was not located.
- `NSExecutableNotLoadableError` - returned if the bundle's executable file exists but could not be loaded. This error is returned if the executable is not recognized as a loadable executable. It can also be returned if the executable is a PEF/CFM executable but the current process does not support that type of executable.
- `NSExecutableArchitectureMismatchError` - returned if the bundle executable does not include code that matches the processor architecture of the current processor.
- `NSExecutableRuntimeMismatchError` - returned if the bundle's required Objective-C runtime information is not compatible with the runtime of the current process.
- `NSExecutableLoadError` - returned if the bundle's executable failed to load for some detectable reason prior to linking. This error might occur if the bundle depends on a framework or library that is missing or if the required framework or library is not compatible with the current architecture or runtime version.
- `NSExecutableLinkError` - returned if the executable failed to load due to link errors but is otherwise alright.

The error object may contain additional debugging information in its description that you can use to identify the cause of the error. (This debugging information should not be displayed to the user.) You can obtain the debugging information by invoking the error object's `description` method in your code or by using the `print-object` command on the error object in `gdb`.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [load](#) (page 24)
- [unload](#) (page 38)

Declared In

`NSBundle.h`

localizations

Returns a list of all the localizations contained within the receiver's bundle.

```
- (NSArray *)localizations
```

Return Value

An array, containing `NSString` objects, that specifies all the localizations contained within the receiver's bundle.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSBundle.h`

localizedInfoDictionary

Returns a dictionary with the keys from the bundle's localized property list.

```
- (NSDictionary *)localizedInfoDictionary
```

Return Value

A dictionary with the keys from the bundle's localized property list (`InfoPlist.strings`).

Discussion

This method uses the preferred localization for the current user when determining which resources to return. If the preferred localization is not available, this method chooses the most appropriate localization found in the bundle.

Availability

Available in Mac OS X v10.2 and later.

Related Sample Code

`PrefsPane`

Declared In

`NSBundle.h`

localizedStringForKey:value:table:

Returns a localized version of the string designated by the specified key and residing in the specified table.

```
- (NSString *)localizedStringForKey:(NSString *)key value:(NSString *)value
    table:(NSString *)tableName
```

Parameters

key

The key for a string in the table identified by *tableName*.

value

The value to return if *key* is `nil` or if a localized string for *key* can't be found in the table.

tableName

The receiver's string table to search. If *tableName* is `nil` or is an empty string, the method attempts to use the table in `Localizable.strings`.

Return Value

A localized version of the string designated by *key* in table *tableName*. If *value* is `nil` or an empty string, and a localized string is not found in the table, returns *key*. If *key* and *value* are both `nil`, returns the empty string.

Discussion

For more details about string localization and the specification of a `.strings` file, see “Working With Localized Strings.”

Using the user default `NSShowNonLocalizedStrings`, you can alter the behavior of [localizedStringForKey:value:table:](#) (page 26) to log a message when the method can't find a localized string. If you set this default to `YES` (in the global domain or in the application's domain), then when the method can't find a localized string in the table, it logs a message to the console and capitalizes *key* before returning it.

The following example cycles through a static array of keys when a button is clicked, gets the value for each key from a strings table named `Buttons.strings`, and sets the button title with the returned value:

```
- (void)changeTitle:(id)sender
{
    static int keyIndex = 0;
   NSBundle *thisBundle = [NSBundle bundleForClass:[self class]];

    NSString *locString = [thisBundle
        localizedStringForKey:assortedKeys[keyIndex++]
        value:@"No translation" table:@"Buttons"];
    [sender setTitle:locString];
    if (keyIndex == MAXSTRINGS) keyIndex=0;
}
```

Availability

Available in Mac OS X v10.0 and later.

See Also

- [pathForResource ofType:](#) (page 29)
- [pathForResource ofType: inDirectory:](#) (page 30)
- [pathsForResourcesOfType: inDirectory:](#) (page 31)
- + [pathForResource ofType: inDirectory:](#) (page 13)
- + [pathsForResourcesOfType: inDirectory:](#) (page 14)

Related Sample Code

BundleLoader

CocoaDVDPlayer

ImageApp

Sketch+Accessibility

Sketch-112

Declared In

NSBundle.h

objectForKey:

Returns the value associated with the specified key in the receiver's information property list.

```
- (id)objectForKey:(NSString *)key
```

Parameters

key

A key in the receiver's property list.

Return Value

The value associated with *key* in the receiver's property list (`Info.plist`). The localized value of a key is returned when one is available.

Discussion

Use of this method is preferred over other access methods because it returns the localized value of a key when one is available.

Availability

Available in Mac OS X v10.2 and later.

Related Sample Code

AutoUpdater

BundleLoader

FancyAbout

GridCalendar

Declared In

NSBundle.h

pathForAuxiliaryExecutable:

Returns the full pathname of the executable with the specified name in the receiver's bundle.

```
- (NSString *)pathForAuxiliaryExecutable:(NSString *)executableName
```

Parameters

executableName

The name of an executable file.

Return Value

The full pathname of the executable *executableName* in the receiver's bundle.

Discussion

This method returns the appropriate path for modern application and framework bundles. This method may not return a path for non-standard bundle formats or for some older bundle formats.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSBundle.h

pathForResource ofType:

Returns the full pathname for the resource identified by the specified name and file extension.

- (NSString *)pathForResource:(NSString *)*name* ofType:(NSString *)*extension*

Parameters

name

The name of the resource file.

extension

If *extension* is an empty string or `nil`, the extension is assumed not to exist and the file URL is the first file encountered that exactly matches *name*.

Return Value

The full pathname for the resource file or `nil` if the file could not be located.

Discussion

The method first looks for a matching resource file in the non-localized resource directory of the specified bundle. (In Mac OS X, this directory is typically called `Resources` but in iPhone OS, it is the main bundle directory.) If a matching resource file is not found, it then looks in the top level of any available language-specific “.lproj” directories. (The search order for the language-specific directories corresponds to the user’s preferences.) It does not recurse through other subdirectories at any of these locations. For more details see Bundles and Localization.

The following code fragment gets the path to a plist within the bundle, and loads it into an `NSDictionary`.

```
NSBundle *thisBundle = [NSBundle bundleForClass:[self class]];
if (commonDictionaryPath = [thisBundle pathForResource:@"CommonDictionary"
ofType:@"plist"]) {
    NSDictionary = [[NSDictionary alloc]
initWithContentsOfFile:commonDictionaryPath];
    // when completed, it is the developer's responsibility to release
theDictionary
}
```

Availability

Available in Mac OS X v10.0 and later.

See Also

- [URLForResource:withExtension:](#) (page 39)

Related Sample Code

AttachAScript

CIAnnotation

FunHouse

GLSLShowpiece

ImageKitDemo

Declared In

NSBundle.h

pathForResource ofType:inDirectory:

Returns the full pathname for the resource identified by the specified name and file extension and located in the specified bundle subdirectory.

```
- (NSString *)pathForResource:(NSString *)name ofType:(NSString *)extension
    inDirectory:(NSString *)subpath
```

Parameters

name

The name of the resource file.

extension

If *extension* is an empty string or `nil`, the extension is assumed not to exist and the file URL is the first file encountered that exactly matches *name*.

subpath

The name of the bundle subdirectory.

Return Value

The full pathname for the resource file or `nil` if the file could not be located.

Discussion

If *subpath* is `nil`, this method searches the top-level nonlocalized resource directory and the top-level of any language-specific directories. (In Mac OS X, the top-level nonlocalized resource directory is typically called `Resources` but in iPhone OS, it is the main bundle directory.) For example, suppose you have a Mac OS X application with a modern bundle and you specify `@ "Documentation"` for the *subpath* parameter. This method would first look in the `Contents/Resources/Documentation` directory of the bundle, followed by the `Documentation` subdirectories of each language-specific `.lproj` directory. (The search order for the language-specific directories corresponds to the user's preferences.) This method does not recurse through any other subdirectories at any of these locations. For more details see [Bundles and Localization](#).

Availability

Available in Mac OS X v10.0 and later.

See Also

- [localizedStringForKey:value:table:](#) (page 26)
- [pathForResource ofType:](#) (page 29)
- [pathsForResourcesOfType:inDirectory:](#) (page 31)
- + [pathForResource ofType:inDirectory:](#) (page 13)
- + [pathsForResourcesOfType:inDirectory:](#) (page 14)

Declared In

`NSBundle.h`

pathForResource ofType:inDirectory:forLocalization:

Returns the full pathname for the resource identified by the specified name and file extension, located in the specified bundle subdirectory, and limited to global resources and those associated with the specified localization.

```
- (NSString *)pathForResource:(NSString *)name ofType:(NSString *)extension
    inDirectory:(NSString *)subpath forLocalization:(NSString *)localizationName
```

Parameters*name*

The name of the resource file.

extension

If *extension* is an empty string or `nil`, the extension is assumed not to exist and the file URL is the first file encountered that exactly matches *name*.

subpath

The name of the bundle subdirectory to search.

localizationName

The name of the localization. This parameter should correspond to the name of one of the bundle's language-specific resource directories without the `.lproj` extension.

Return Value

The full pathname for the resource file or `nil` if the file could not be located.

Discussion

This method is equivalent to [pathForResource ofType:inDirectory:](#) (page 30), except that only nonlocalized resources and those in the language-specific `.lproj` directory specified by *localizationName* are searched.

There should typically be little reason to use this method—see [Getting the Current Language and Locale](#). See also [preferredLocalizationsFromArray:forPreferences:](#) (page 16) for how to determine what localizations are available.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSBundle.h

pathsForResourceOfType:inDirectory:

Returns an array containing the pathnames for all bundle resources having the specified filename extension and residing in the resource subdirectory.

```
- (NSArray *)pathsForResourceOfType:(NSString *)extension inDirectory:(NSString *)subpath
```

Parameters*extension*

If *extension* is an empty string or `nil`, the extension is assumed not to exist and the file URL is the first file encountered that exactly matches *name*.

subpath

The name of the bundle subdirectory to search.

Return Value

An array containing the full pathnames for all bundle resources matching the specified criteria. This method returns an empty array if no matching resource files are found.

Discussion

This method provides a means for dynamically discovering multiple bundle resources of the same type. If *extension* is an empty string or `nil`, all bundle resources in the specified resource directory are returned.

The argument *subpath* specifies the name of a specific subdirectory to search within the current bundle's resource directory hierarchy. If *subpath* is `nil`, this method searches the top-level nonlocalized resource directory and the top-level of any language-specific directories. (In Mac OS X, the top-level nonlocalized resource directory is typically called `Resources` but in iPhone OS, it is the main bundle directory.) For example, suppose you have a Mac OS X application with a modern bundle and you specify `@\"Documentation\"` for the *subpath* parameter. This method would first look in the `Contents/Resources/Documentation` directory of the bundle, followed by the `Documentation` subdirectories of each language-specific `.lproj` directory. (The search order for the language-specific directories corresponds to the user's preferences.) This method does not recurse through any other subdirectories at any of these locations. For more details see [Bundles and Localization](#).

Availability

Available in Mac OS X v10.0 and later.

See Also

- [localizedStringForKey:value:table:](#) (page 26)
- [pathForResource ofType:](#) (page 29)
- [pathForResource ofType:inDirectory:](#) (page 30)
- + [pathForResource ofType:inDirectory:](#) (page 13)
- + [pathsForResourcesOfType:inDirectory:](#) (page 14)

Related Sample Code

[UIKitCreateMovie](#)

Declared In

`NSBundle.h`

pathsForResourcesOfType:inDirectory:forLocalization:

Returns an array containing the file URLs for all bundle resources having the specified filename extension, residing in the specified resource subdirectory, and limited to global resources and those associated with the specified localization.

```
- (NSArray *)pathsForResourcesOfType:(NSString *)extension inDirectory:(NSString *)subpath forLocalization:(NSString *)localizationName
```

Parameters

extension

The file extension of the files to retrieve.

subpath

The name of the bundle subdirectory to search.

localizationName

The name of the localization. This parameter should correspond to the name of one of the bundle's language-specific resource directories without the `.lproj` extension.

Return Value

An array containing the full pathnames for all bundle resources matching the specified criteria. This method returns an empty array if no matching resource files are found.

Discussion

This method is equivalent to [pathsForResourceOfType:inDirectory:](#) (page 31), except that only nonlocalized resources and those in the language-specific `.lproj` directory specified by *localizationName* are searched.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSBundle.h

preferredLocalizations

Returns an array of strings indicating the actual localizations contained in the receiver's bundle.

- (NSArray *)preferredLocalizations

Return Value

An array of `NSString` objects, each of which identifies the a localization in the receiver's bundle. The localizations in the array are not returned in any particular order.

Availability

Available in Mac OS X v10.0 and later.

See Also

+ [preferredLocalizationsFromArray:](#) (page 15)

- [localizations](#) (page 26)

Declared In

NSBundle.h

preflightAndReturnError:

Returns a Boolean value indicating whether the bundle's executable code could be loaded successfully.

- (BOOL)preflightAndReturnError:(NSError **)error

Parameters

error

On input, a pointer to an error object variable. On output, this variable may contain an error object indicating why the bundle's executable could not be loaded. If no error would occur, this parameter is left unmodified. You may specify `nil` for this parameter if you are not interested in the error information.

Return Value

YES if the bundle's executable code could be loaded successfully or is already loaded; otherwise, NO if the code could not be loaded.

Discussion

This method does not actually load the bundle's executable code. Instead, it performs several checks to see if the code could be loaded and with one exception returns the same errors that would occur during an actual load operation. The one exception is the `NSExecutableLinkError` error, which requires the actual loading of the code to verify link errors.

For a list of possible load errors, see the discussion for the [loadAndReturnError:](#) (page 25) method.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [loadAndReturnError:](#) (page 25)

Declared In

NSBundle.h

principalClass

Returns the receiver's principal class.

```
- (Class)principalClass
```

Return Value

The receiver's principal class—after ensuring that the code containing the definition of that class is dynamically loaded. If the receiver encounters errors in loading or if it can't find the executable code file in the bundle directory, returns `nil`.

Discussion

The principal class typically controls all the other classes in the bundle; it should mediate between those classes and classes external to the bundle. Classes (and categories) are loaded from just one file within the bundle directory. `NSBundle` obtains the name of the code file to load from the dictionary returned from [infoDictionary](#) (page 22), using “NSExecutable” as the key. The bundle determines its principal class in one of two ways:

- It first looks in its own information dictionary, which extracts the information encoded in the bundle's property list (`Info.plist`). `NSBundle` obtains the principal class from the dictionary using the key `NSPrincipalClass`. For non-loadable bundles (applications and frameworks), if the principal class is not specified in the property list, the method returns `nil`.
- If the principal class is not specified in the information dictionary, `NSBundle` identifies the first class loaded as the principal class. When several classes are linked into a dynamically loadable file, the default principal class is the first one listed on the `ld` command line. In the following example, `Reporter` would be the principal class:

```
ld -o myBundle -r Reporter.o NotePad.o QueryList.o
```

The order of classes in Xcode's project browser is the order in which they will be linked. To designate the principal class, control-drag the file containing its implementation to the top of the list.

As a side effect of code loading, the receiver posts [NSBundleDidLoadNotification](#) (page 44) after all classes and categories have been loaded; see “[Notifications](#)” (page 44) for details.

The following method obtains a bundle by specifying its path ([bundleWithPath:](#) (page 11)), then loads the bundle with [principalClass](#) (page 34) and uses the returned class object to allocate and initialize an instance of that class:

```
- (void)loadBundle:(id)sender
{
    Class exampleClass;
```

```

    id newInstance;
    NSString *path = @"/tmp/Projects/BundleExample/BundleExample.bundle";
   NSBundle *bundleToLoad = [NSBundle bundleWithPath:path];
    if (exampleClass = [bundleToLoad principalClass]) {
        newInstance = [[exampleClass alloc] init];
        [newInstance doSomething];
    }
}

```

Availability

Available in Mac OS X v10.0 and later.

See Also

- [classNameed:](#) (page 20)
- [infoDictionary](#) (page 22)
- [load](#) (page 24)

Related Sample Code

BundleLoader

Declared In

NSBundle.h

privateFrameworksPath

Returns the full pathname of the receiver's subdirectory containing private frameworks.

```
- (NSString *)privateFrameworksPath
```

Return Value

The full pathname of the receiver's subdirectory containing private frameworks.

Discussion

This method returns the appropriate path for modern application and framework bundles. This method may not return a path for non-standard bundle formats or for some older bundle formats.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

MP3 Player

Declared In

NSBundle.h

privateFrameworksURL

Returns the file URL of the receiver's subdirectory containing private frameworks.

```
- (NSURL *)privateFrameworksURL
```

Return Value

The file URL of the receiver's subdirectory containing private frameworks.

Discussion

This method returns the appropriate path for modern application and framework bundles. This method may not return a URL for non-standard bundle formats or for some older bundle formats.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSBundle.h

resourcePath

Returns the full pathname of the receiving bundle's subdirectory containing resources.

- (NSString *)resourcePath

Return Value

The full pathname of the receiving bundle's subdirectory containing resources.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [bundlePath](#) (page 19)

Related Sample Code

FunHouse

SimpleStickies

SurfaceVertexProgram

TextureRange

VertexPerformanceDemo

Declared In

NSBundle.h

resourceURL

Returns the file URL of the receiver's subdirectory containing resource files.

- (NSURL *)resourceURL

Return Value

The file URL of the receiver's subdirectory containing resource files.

Discussion

This method returns the appropriate path for modern application and framework bundles. This method may not return a path for non-standard bundle formats or for some older bundle formats.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSBundle.h

sharedFrameworksPath

Returns the full pathname of the receiver's subdirectory containing shared frameworks.

- (NSString *)sharedFrameworksPath

Return Value

The full pathname of the receiver's subdirectory containing shared frameworks.

Discussion

This method returns the appropriate path for modern application and framework bundles. This method may not return a path for non-standard bundle formats or for some older bundle formats.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSBundle.h

sharedFrameworksURL

Returns the file URL of the receiver's subdirectory containing shared frameworks.

- (NSURL *)sharedFrameworksURL

Return Value

The file URL of the receiver's subdirectory containing shared frameworks.

Discussion

This method returns the appropriate path for modern application and framework bundles. This method may not return a URL for non-standard bundle formats or for some older bundle formats.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSBundle.h

sharedSupportPath

Returns the full pathname of the receiver's subdirectory containing shared support files.

- (NSString *)sharedSupportPath

Return Value

The full pathname of the receiver's subdirectory containing shared support files.

Discussion

This method returns the appropriate path for modern application and framework bundles. This method may not return a path for non-standard bundle formats or for some older bundle formats.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSBundle.h

sharedSupportURL

Returns the file URL of the receiver's subdirectory containing shared support files.

- (NSURL *)sharedSupportURL

Return Value

The file URL of the receiver's subdirectory containing shared support files.

Discussion

This method returns the appropriate path for modern application and framework bundles. This method may not return a path for non-standard bundle formats or for some older bundle formats.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSBundle.h

unload

Unloads the code associated with the receiver.

- (BOOL)unload

Return Value

YES if the bundle was successfully unloaded or was not already loaded; otherwise, NO if the bundle could not be unloaded.

Discussion

This method attempts to unload a bundle's executable code using the underlying dynamic loader (typically `dyld`). You may use this method to unload plug-in and framework bundles when you no longer need the code they contain. You should use this method to unload bundles that were loaded using the methods of the `NSBundle` class only. Do not use this method to unload bundles that were originally loaded using the bundle-manipulation functions in Core Foundation.

It is the responsibility of the caller to ensure that no in-memory objects or data structures refer to the code being unloaded. For example, if you have an object whose class is defined in a bundle, you must release that object prior to unloading the bundle. Similarly, your code should not attempt to access any symbols defined in an unloaded bundle.

Special Considerations

Prior to Mac OS X version 10.5, code could not be unloaded once loaded, and this method would always return NO. In Mac OS X version 10.5 and later, you can unload a bundle's executable code using this method.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [loadAndReturnError:](#) (page 25)

- [load](#) (page 24)

Declared In

NSBundle.h

URLForAuxiliaryExecutable:

Returns the file URL of the executable with the specified name in the receiver's bundle.

```
- (NSURL *)URLForAuxiliaryExecutable:(NSString *)executableName
```

Parameters

executableName

The name of an executable file.

Return Value

The file URL of the executable *executableName* in the receiver's bundle.

Discussion

This method returns the appropriate path for modern application and framework bundles. This method may not return a URL for non-standard bundle formats or for some older bundle formats.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSBundle.h

URLForResource:withExtension:

Returns the file URL for the resource identified by the specified name and file extension.

```
- (NSURL *)URLForResource:(NSString *)name withExtension:(NSString *)extension
```

Parameters

name

The name of the resource file.

extension

If *extension* is an empty string or `nil`, the extension is assumed not to exist and the file URL is the first file encountered that exactly matches *name*.

Return Value

The file URL for the resource file or `nil` if the file could not be located.

Discussion

If *extension* is an empty string or `nil`, the returned pathname is the first one encountered where the file name exactly matches *name*.

The method first looks for a matching resource file in the nonlocalized resource directory of the specified bundle. (In Mac OS X, this directory is typically called `Resources` but in iPhone OS, it is the main bundle directory.) If a matching resource file is not found, it then looks in the top level of any available

language-specific “.lproj” directories. (The search order for the language-specific directories corresponds to the user’s preferences.) It does not recurse through other subdirectories at any of these locations. For more details see Bundles and Localization.

Availability

Available in Mac OS X v10.6 and later.

Related Sample Code

DispatchFractal

TextSizingExample

Declared In

NSBundle.h

URLForResource:withExtension:subdirectory:

Returns the file URL for the resource file identified by the specified name and extension and residing in a given bundle directory.

```
- (NSURL *)URLForResource:(NSString *)name withExtension:(NSString *)extension
subdirectory:(NSString *)subpath
```

Parameters

name

The name of a resource file contained in the directory specified by *bundleURL*.

extension

If *extension* is an empty string or *nil*, the extension is assumed not to exist and the file URL is the first file encountered that exactly matches *name*.

subpath

The path of a top-level bundle directory. This must be a valid path. For example, to specify the bundle directory for a Mac OS X application, you might specify the path `/Applications/MyApp.app`.

Return Value

The file URL for the resource file or *nil* if the file could not be located. This method also returns *nil* if the bundle specified by the `bundlePath` parameter does not exist or is not a readable directory.

Discussion

The method first looks for a matching resource file in the non-localized resource directory of the specified bundle. (In Mac OS X, this directory is typically called `Resources` but in iPhone OS, it is the main bundle directory.) If a matching resource file is not found, it then looks in the top level of any available language-specific “.lproj” directories. (The search order for the language-specific directories corresponds to the user’s preferences.) It does not recurse through other subdirectories at any of these locations. For more details see Bundles and Localization.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [pathForResource ofType:inDirectory:](#) (page 30)

Declared In

NSBundle.h

URLForResource:withExtension:subdirectory:localization:

Returns the file URL for the resource identified by the specified name and file extension, located in the specified bundle subdirectory, and limited to global resources and those associated with the specified localization.

```
- (NSURL *)URLForResource:(NSString *)name withExtension:(NSString *)extension
    subdirectory:(NSString *)subpath localization:(NSString *)localizationName
```

Parameters

name

The name of the resource file.

extension

If *extension* is an empty string or `nil`, the extension is assumed not to exist and the file URL is the first file encountered that exactly matches *name*.

subpath

The name of the bundle subdirectory to search.

localizationName

The name of the localization. This parameter should correspond to the name of one of the bundle's language-specific resource directories without the `.lproj` extension.

Return Value

The file URL for the resource file or `nil` if the file could not be located.

Discussion

This method is equivalent to [URLsForResourceWithExtension:subdirectory:](#) (page 41), except that only nonlocalized resources and those in the language-specific `.lproj` directory specified by *localizationName* are searched.

There should typically be little reason to use this method—see [Getting the Current Language and Locale](#). See also [preferredLocalizationsFromArray:forPreferences:](#) (page 16) for how to determine what localizations are available.

Availability

Available in Mac OS X v10.6 and later.

Declared In

NSBundle.h

URLsForResourceWithExtension:subdirectory:

Returns the file URL for the resource identified by the specified name and file extension and located in the specified bundle subdirectory.

```
- (NSArray *)URLsForResourceWithExtension:(NSString *)extension
    subdirectory:(NSString *)subpath
```

Parameters

name

The name of the resource file.

extension

If *extension* is an empty string or `nil`, the extension is assumed not to exist and the file URL is the first file encountered that exactly matches *name*.

subpath

The name of the bundle subdirectory.

Return Value

The file URL for the resource file or `nil` if the file could not be located.

Discussion

If *subpath* is `nil`, this method searches the top-level non-localized resource directory and the top-level of any language-specific directories. (In Mac OS X, the top-level non-localized resource directory is typically called `Resources` but in iPhone OS, it is the main bundle directory.) For example, suppose you have a Mac OS X application with a modern bundle and you specify `@ "Documentation"` for the *subpath* parameter. This method would first look in the `Contents/Resources/Documentation` directory of the bundle, followed by the `Documentation` subdirectories of each language-specific `.lproj` directory. (The search order for the language-specific directories corresponds to the user's preferences.) This method does not recurse through any other subdirectories at any of these locations. For more details see Bundles and Localization.

Availability

Available in Mac OS X v10.6 and later.

Declared In

`NSBundle.h`

URLsForResourcesWithExtension:subdirectory:localization:

Returns an array containing the file URLs for all bundle resources having the specified filename extension, residing in the specified resource subdirectory, and limited to global resources and those associated with the specified localization.

```
- (NSArray *)URLsForResourcesWithExtension:(NSString *)extensions
    subdirectory:(NSString *)subpath localization:(NSString *)localizationName
```

Parameters

ext

The file extension of the files to retrieve.

subpath

The name of the bundle subdirectory to search.

localizationName

The name of the localization. This parameter should correspond to the name of one of the bundle's language-specific resource directories without the `.lproj` extension.

Return Value

An array containing the file URLs for all bundle resources matching the specified criteria. This method returns an empty array if no matching resource files are found.

Discussion

This method is equivalent to `URLsForResourcesWithExtension:subdirectory:` (page 41), except that only nonlocalized resources and those in the language-specific `.lproj` directory specified by *localizationName* are searched.

Availability

Available in Mac OS X v10.6 and later.

Declared In

`NSBundle.h`

Constants

Mach-O Architecture

These constants describe the CPU types that a bundle's executable code may support.

```
enum {
    NSBundleExecutableArchitectureI386      = 0x00000007,
    NSBundleExecutableArchitecturePPC      = 0x00000012,
    NSBundleExecutableArchitectureX86_64   = 0x01000007,
    NSBundleExecutableArchitecturePPC64    = 0x01000012
};
```

Constants

`NSBundleExecutableArchitectureI386`

Specifies the 32-bit Intel architecture.

Available in Mac OS X v10.5 and later.

Declared in `NSBundle.h`.

`NSBundleExecutableArchitecturePPC`

Specifies the 32-bit PowerPC architecture.

Available in Mac OS X v10.5 and later.

Declared in `NSBundle.h`.

`NSBundleExecutableArchitectureX86_64`

Specifies the 64-bit Intel architecture.

Available in Mac OS X v10.5 and later.

Declared in `NSBundle.h`.

`NSBundleExecutableArchitecturePPC64`

Specifies the 64-bit PowerPC architecture.

Available in Mac OS X v10.5 and later.

Declared in `NSBundle.h`.

NSLoadedClasses

This constant is provided in the `userInfo` dictionary of the `NSBundleDidLoadNotification` (page 44) notification.

```
NSString * const NSLoadedClasses;
```

Constants

`NSLoadedClasses`

An `NSArray` object containing the names (as `NSString` objects) of each class that was loaded

Available in Mac OS X v10.0 and later.

Declared in `NSBundle.h`.

Notifications

NSBundleDidLoadNotification

`NSBundle` posts `NSBundleDidLoadNotification` to notify observers which classes and categories have been dynamically loaded. When a request is made to an `NSBundle` object for a class (`classNameed:` (page 20) or `principalClass` (page 34)), the bundle dynamically loads the executable code file that contains the class implementation and all other class definitions contained in the file. After the module is loaded, the bundle posts the `NSBundleDidLoadNotification`.

The notification object is the `NSBundle` instance that dynamically loads classes. The `userInfo` dictionary contains an `NSLoadedClasses` (page 43) key.

In a typical use of this notification, an object might want to enumerate the `userInfo` array to check if each loaded class conformed to a certain protocol (say, an protocol for a plug-and-play tool set); if a class does conform, the object would create an instance of that class and add the instance to another `NSArray` object.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSBundle.h`

Document Revision History

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

Date	Notes
2009-07-23	Updated for Mac OS X v10.6. New URL methods.
2009-05-12	Updated method descriptions to reflect iPhone bundle directory structure.
2007-07-19	Updated for Mac OS X v10.5.
2006-05-23	First publication of this content as a separate document.

REVISION HISTORY

Document Revision History

Index

A

`allBundles` **class method** [9](#)
`allFrameworks` **class method** [10](#)

B

`builtinPlugInsPath` **instance method** [18](#)
`builtinPlugInsURL` **instance method** [18](#)
`bundleForClass:` **class method** [10](#)
`bundleIdentifier` **instance method** [18](#)
`bundlePath` **instance method** [19](#)
`bundleURL` **instance method** [19](#)
`bundleWithIdentifier:` **class method** [11](#)
`bundleWithPath:` **class method** [11](#)
`bundleWithURL:` **class method** [12](#)

C

`classNameed:` **instance method** [20](#)

D

`developmentLocalization` **instance method** [20](#)

E

`executableArchitectures` **instance method** [21](#)
`executablePath` **instance method** [21](#)
`executableURL` **instance method** [21](#)

I

`infoDictionary` **instance method** [22](#)
`initWithPath:` **instance method** [22](#)
`initWithURL:` **instance method** [23](#)
`isLoading` **instance method** [24](#)

L

`load` **instance method** [24](#)
`loadAndReturnError:` **instance method** [25](#)
`localizations` **instance method** [26](#)
`localizedInfoDictionary` **instance method** [26](#)
`localizedStringForKey:value:table:` **instance method** [26](#)

M

`Mach-O Architecture` [43](#)
`mainBundle` **class method** [13](#)

N

`NSBundleDidLoadNotification` **notification** [44](#)
`NSBundleExecutableArchitectureI386` **constant** [43](#)
`NSBundleExecutableArchitecturePPC` **constant** [43](#)
`NSBundleExecutableArchitecturePPC64` **constant** [43](#)
`NSBundleExecutableArchitectureX86_64` **constant** [43](#)
`NSLoadedClasses` [43](#)
`NSLoadedClasses` **constant** [43](#)

O

`objectForInfoDictionaryKey:` **instance method** [28](#)

P

pathForAuxiliaryExecutable: **instance method** [28](#)
 pathForResource:ofType: **instance method** [29](#)
 pathForResource:ofType:inDirectory: **class method** [13](#)
 pathForResource:ofType:inDirectory: **instance method** [30](#)
 pathForResource:ofType:inDirectory:
 forLocalization: **instance method** [30](#)
 pathsForResourcesOfType:inDirectory: **class method** [14](#)
 pathsForResourcesOfType:inDirectory: **instance method** [31](#)
 pathsForResourcesOfType:inDirectory:
 forLocalization: **instance method** [32](#)
 preferredLocalizations **instance method** [33](#)
 preferredLocalizationsFromArray: **class method** [15](#)
 preferredLocalizationsFromArray:forPreferences:
 class method [16](#)
 preflightAndReturnError: **instance method** [33](#)
 principalClass **instance method** [34](#)
 privateFrameworksPath **instance method** [35](#)
 privateFrameworksURL **instance method** [35](#)

URLForResource:withExtension:subdirectory:
 localization: **instance method** [41](#)
 URLsForResourcesWithExtension:subdirectory:
 instance method [41](#)
 URLsForResourcesWithExtension:subdirectory:
 inBundleWithURL: **class method** [17](#)
 URLsForResourcesWithExtension:subdirectory:
 localization: **instance method** [42](#)

R

resourcePath **instance method** [36](#)
 resourceURL **instance method** [36](#)

S

sharedFrameworksPath **instance method** [37](#)
 sharedFrameworksURL **instance method** [37](#)
 sharedSupportPath **instance method** [37](#)
 sharedSupportURL **instance method** [38](#)

U

unload **instance method** [38](#)
 URLForAuxiliaryExecutable: **instance method** [39](#)
 URLForResource:withExtension: **instance method** [39](#)
 URLForResource:withExtension:subdirectory:
 instance method [40](#)
 URLForResource:withExtension:subdirectory:
 inBundleWithURL: **class method** [16](#)