
UINib Class Reference

Data Management: File Management



2010-05-26



Apple Inc.
© 2010 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, and iPhone are trademarks of Apple Inc., registered in the United States and other countries.

iOS is a trademark or registered trademark of Cisco in the U.S. and other countries and is used under license.

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

UINavigationController Class Reference 5

Overview 5

Tasks 5

 Creating a UINavigationController Object 5

 Instantiating a UINavigationController 6

Class Methods 6

 initWithData:bundle: 6

 initWithNibName:bundle: 6

Instance Methods 7

 initWithOwner:options: 7

Document Revision History 9

UINib Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/UIKit.framework
Availability	Available in iOS 4.0 and later.
Declared in	UINib.h

Overview

Instances of the `UINib` class serve as object wrappers, or containers, for Interface Builder nib files. An `UINib` object caches the contents of a nib file in memory, ready for unarchiving and instantiation. When your application needs to instantiate the contents of the nib file it can do so without having to load the data from the nib file first, improving performance. The `UINib` object can automatically release this cached nib data to free up memory for your application under low-memory conditions, reloading that data the next time your application instantiates the nib. Your application should use `UINib` objects whenever it needs to repeatedly instantiate the same nib data. For example, if your table view uses a nib file to instantiate table view cells, caching the nib in a `UINib` object can provide a significant performance improvement.

When you create an `UINib` object using the contents of a nib file, the object loads the object graph in the referenced nib file, but it does not yet unarchive it. To unarchive all of the nib data and thus truly instantiate the nib your application calls the `instantiateWithOwner:options:` method on the `UINib` object. The steps that the `UINib` object follows to instantiate the nib's object graph are described in detail in *Resource Programming Guide*.

Tasks

Creating a Nib Object

- + `nibWithName:bundle:` (page 6)
Returns an `UINib` object initialized to the nib file in the specified bundle.
- + `nibWithData:bundle:` (page 6)
Creates an `UINib` object from nib data stored in memory.

Instantiating a Nib

- [initWithOwner:options:](#) (page 7)

Unarchives and instantiates the in-memory contents of the receiver's nib file, creating a distinct object tree and set of top level objects.

Class Methods

nibWithData:bundle:

Creates an UINib object from nib data stored in memory.

```
+ (UINib *)nibWithData:(NSData *)data bundle:(NSBundle *)bundleOrNil
```

Parameters

data

A block of memory that contains nib data.

bundleOrNil

The bundle in which to search for resources referenced by the nib. If you specify `nil`, this method looks for the nib file in the main bundle.

Return Value

The initialized `NSNib` object or `nil` if there were errors during initialization.

Discussion

The UINib object looks for the nib file in the bundle's language-specific project directories first, followed by the `Resources` directory.

The preferred mechanism for instantiating UINib objects is with the `nibWithNibName:bundle:` class method. A UINib object instantiated using the `nibWithData:bundle:` class method cannot release the cached data under low memory conditions. Your application should be prepared to release the UINib object and the data under low memory conditions, recreating both the next time the application needs to instantiate the nib.

Availability

Available in iOS 4.0 and later.

Declared In

UINib.h

nibWithNibName:bundle:

Returns an UINib object initialized to the nib file in the specified bundle.

```
+ (UINib *)nibWithNibName:(NSString *)name bundle:(NSBundle *)bundleOrNil
```

Parameters

name

The name of the nib file, without any leading path information.

bundleOrNil

The bundle in which to search for the nib file. If you specify `nil`, this method looks for the nib file in the main bundle.

Return Value

The initialized `UINib` object or `nil` if there were errors during initialization or the nib file could not be located.

Discussion

The `UINib` object looks for the nib file in the bundle's language-specific project directories first, followed by the `Resources` directory.

Availability

Available in iOS 4.0 and later.

Declared In

`UINib.h`

Instance Methods

instantiateWithOwner:options:

Unarchives and instantiates the in-memory contents of the receiver's nib file, creating a distinct object tree and set of top level objects.

```
- (NSArray *)instantiateWithOwner:(id)ownerOrNil options:(NSDictionary *)optionsOrNil
```

Parameters*ownerOrNil*

The object to use as the owner of the nib file. If the nib file has an owner, you must specify a valid object for this parameter.

optionsOrNil

A dictionary containing the options to use when opening the nib file. For a list of available keys for this dictionary, see *NSBundle UIKit Additions Reference*.

Return Value

An autoreleased `NSArray` object containing the top-level objects from the nib file.

Discussion

You can use this method to instantiate the objects in a nib and provide them to your code. This method unarchives each object, initializes it, sets its properties to their configured values, and reestablishes any connections to other objects. For detailed information about the nib-loading process, see *Resource Programming Guide*.

If the nib file contains any proxy objects beyond just the File's Owner proxy object, you can specify the runtime replacement objects for those proxies using the options dictionary. In that dictionary, add the `UINibExternalObjects` key and set its value to a dictionary containing the names of any proxy objects (the keys) and the real objects to use in their place. The proxy object's name is the string you assign to it in the Name field of the Interface Builder inspector window.

Availability

Available in iOS 4.0 and later.

Declared In
UINib.h

Document Revision History

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

Date	Notes
2010-05-26	New document that describes an object for unarchiving and instantiating a nib file.

REVISION HISTORY

Document Revision History