NSNotification Class Reference

Cocoa > Events & Other Input

2007-04-02

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

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

NSNotification Class Reference

Inherits from	NSObject
Conforms to	NSCoding NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/Foundation.framework
Availability	Available in Mac OS X v10.0 and later.
Companion guide	Notification Programming Topics for Cocoa
Declared in	NSNotification.h
Related sample code	EnhancedAudioBurn PDFKitLinker2 Quartz Composer WWDC 2005 TextEdit Sketch-112 TextEditPlus

Overview

NSNotification objects encapsulate information so that it can be broadcast to other objects by an NSNotificationCenter object. An NSNotification object (referred to as a notification) contains a name, an object, and an optional dictionary. The name is a tag identifying the notification. The object is any object that the poster of the notification wants to send to observers of that notification (typically, it is the object that posted the notification). The dictionary stores other related objects, if any. NSNotification objects are immutable objects.

You can create a notification object with the class methods `notificationWithName:object:` (page 7) or `notificationWithName:object:userInfo:` (page 7). However, you don't usually create your own notifications directly. The NSNotificationCenter methods `postNotificationName:object:` and `postNotificationName:object:userInfo:` allow you to conveniently post a notification without creating it first.

NSCopying Protocol

The NSNotification class adopts the NSCopying protocol, making it possible to treat notifications as context-independent values that can be copied and reused. You can store a notification for later use or use the distributed objects system to send a notification to another process. The NSCopying protocol essentially allows clients to deal with notifications as first class values that can be copied by collections. You can put notifications in an array and send the `copy` message to that array, which recursively copies every item.

Creating Subclasses

You can subclass NSNotification to contain information in addition to the notification name, object, and dictionary. This extra data must be agreed upon between notifiers and observers.

NSNotification is a class cluster with no instance variables. As such, you must subclass NSNotification and override the primitive methods `name` (page 8), `object` (page 8), and `userInfo` (page 9). You can choose any designated initializer you like, but be sure that your initializer does not call NSNotification's implementation of `init` (via `[super init]`). NSNotification is not meant to be instantiated directly, and its `init` method raises an exception.

Adopted Protocols

NSCoding

- `- encodeWithCoder:`
- `- initWithCoder:`

NSCopying

- `- copyWithZone:`

Tasks

Creating Notifications

`+ notificationWithName:object:` (page 7)
Returns a new notification object with a specified name and object.

`+ notificationWithName:object:userInfo:` (page 7)
Returns a notification object with a specified name, object, and user information.

Getting Notification Information

`- name` (page 8)
Returns the name of the notification.

`- object` (page 8)
Returns the object associated with the notification.

`- userInfo` (page 9)
Returns the user information dictionary associated with the receiver.

Class Methods

notificationWithName:object:

Returns a new notification object with a specified name and object.

```
+ (id)notificationWithName:(NSString *)aName object:(id)anObject
```

Parameters

aName
The name for the new notification. May not be `nil`.

anObject
The object for the new notification.

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

See Also
`- postNotificationName:object:` (NSNotificationCenter)

Related Sample Code
ExtractMovieAudioToAIFF

Link Snoop

MyPhoto

QTExtractAndConvertToAIFF

QTExtractAndConvertToMovieFile

Declared In
`NSNotification.h`

notificationWithName:object:userInfo:

Returns a notification object with a specified name, object, and user information.

```
+ (id)notificationWithName:(NSString *)aName object:(id)anObject
    userInfo:(NSDictionary *)userInfo
```

Parameters

aName
The name for the new notification. May not be `nil`.

anObject
The object for the new notification.

userInfo
  The user information dictionary for the new notification. May be `nil`.

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

See Also
`+ notificationWithName:object:` (page 7)
`- postNotificationName:object:userInfo:` (NSNotificationCenter)

Related Sample Code
People

Declared In
`NSNotification.h`

Instance Methods

name

Returns the name of the notification.

```
- (NSString *)name
```

Return Value
The name of the notification. Typically you use this method to find out what kind of notification you are dealing with when you receive a notification.

Special Considerations

Notification names can be any string. To avoid name collisions, you might want to use a prefix that's specific to your application.

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

Related Sample Code
CoreRecipes
People
QTAudioExtractionPanel
WhackedTV

Declared In
`NSNotification.h`

object

Returns the object associated with the notification.

```
- (id)object
```

Return Value

The object associated with the notification. This is often the object that posted this notification. It may be `nil`.

Typically you use this method to find out what object a notification applies to when you receive a notification.

Discussion

For example, suppose you've registered an object to receive the message `handlePortDeath:` when the "PortInvalid" notification is posted to the notification center and that `handlePortDeath:` needs to access the object monitoring the port that is now invalid. `handlePortDeath:` can retrieve that object as shown here:

```
- (void)handlePortDeath:(NSNotification *)notification
{
    ...
    [self reclaimResourcesForPort:[notification object]];
    ...
}
```

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

ABPresence

NewsReader

Quartz Composer WWDC 2005 TextEdit

Sketch-112

TextEditPlus

Declared In

`NSNotification.h`

userInfo

Returns the user information dictionary associated with the receiver.

```
- (NSDictionary *)userInfo
```

Return Value

Returns the user information dictionary associated with the receiver. May be `nil`.

The user information dictionary stores any additional objects that objects receiving the notification might use.

Discussion

For example, in the Application Kit, NSControl objects post the `NSControlTextDidChangeNotification` whenever the field editor (an NSText object) changes text inside the NSControl. This notification provides the NSControl object as the notification's associated object. In order to provide access to the field editor, the NSControl object posting the notification adds the field editor to the notification's user information dictionary. Objects receiving the notification can access the field editor and the NSControl object posting the notification as follows:

```
- (void)controlTextDidBeginEditing:(NSNotification *)notification
{
    NSText *fieldEditor = [[notification userInfo]
        objectForKey:@"NSFieldEditor"];               // the field editor
    NSControl *postingObject = [notification object]; // the object that posted
 the notification
    ...
}
```

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

Related Sample Code
ABPresence
CoreRecipes
PDFKitLinker2
SimpleCalendar
WhackedTV

Declared In
`NSNotification.h`

Document Revision History

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

Date	Notes
2007-04-02	Made editorial improvements.
2006-05-23	First publication of this content as a separate document.

Index