NSNotificationCenter

Inherits From: NSObject

Conforms To: NSObject (NSObject)

Declared In: Foundation/NSNotification.h

Class at a Glance

Purpose

NSNotificationCenter provides a way for objects that don't know about each other to communicate. It receives NSNotification objects and broadcasts them to all interested objects.

Principal Attributes

A table of objects that want to receive notifications, the notifications they want to receive, and about which objects.

Creation

Each task has a default notification center. You typically don't create your own.

Commonly Used Methods

+ defaultCenter	Accesses the default notification center.
– addObserver:selector:name:object:	Registers an object to receive a notification.
– postNotificationName:object:	Creates and posts a notification.
– removeObserver:	Specifies that an object no longer wants to receive notifications.

Class Description

An NSNotificationCenter object (or simply, *notification center*) is essentially a notification dispatch table. It notifies all observers of notifications meeting specific criteria. This information is encapsulated in NSNotification objects, also known as *notifications*. Client objects register themselves as observers of specific notifications posted by other objects. When an event occurs, an object posts an appropriate notification to the notification center. (See the NSNotification class specification for more on notifications.)

The notification center dispatches a message to each registered observer, passing the notification as the sole argument. It is possible for the posting object and the observing object to be the same.

Each task has a default notification center that you access with the **defaultCenter** class method.

Registering to Receive Notifications

An object registers itself to receive a notification by sending the **addObserver:selector:name:object:** method, specifying the message the notification should send, the name of the notification it wants to receive, and about which object. However, the observer need not specify both the name and the object. If it specifies only the object, it will receive *all* notifications containing that object. If the object specifies only a notification name, it will receive that notification every time it's posted, regardless of the object associated with it.

It is possible for an observer to register to receive more than one message for the same notification. In such a case, the observer will receive all messages it is registered to receive for the notification, but the order in which it receives them cannot be determined.

Creating and Posting Notifications

Normally, you create an instance of NSNotification and post it using **postNotification:**. The methods **postNotificationName:object:** and **postNotificationName:object:userInfo:** are convenient ways to post notifications without having to create an NSNotification first.

Example

As an example of using the notification center, suppose your program can perform a number of conversions on text (for instance, RTF to ASCII). You have defined a class of objects that perform those conversions, Converter. Converter objects might be added or removed during program execution. Your program has a client object that wants to be notified when converters are added or removed, allowing the application to reflect the available services in a pop-up list. The client object would register itself as an observer by sending the following messages to the notification center:

```
[[NSNotificationCenter defaultCenter] addObserver:self
    selector:@selector(objectAddedToConverterList:)
    name:@"ConverterAdded" object:nil];

[[NSNotificationCenter defaultCenter] addObserver:self
    selector:@selector(objectRemovedFromConverterList:)
    name:@"ConverterRemoved" object:nil];
```

When a user installs or removes a Converter, the Converter sends one of the following messages to the notification center:

```
[[NSNotificationCenter defaultCenter]
    postNotificationName:@"ConverterAdded" object:self];
```

or

```
[[NSNotificationCenter defaultCenter]
    postNotificationName:@"ConverterRemoved" object:self];
```

The notification center identifies all observers who are interested in the “ConverterAdded” or “ConverterRemoved” notifications by invoking the method they specified in the selector argument of **addObserver:selector:name:object:**. In the case of our example observer, the selectors are **objectAddedToConverterList:** and **objectRemovedFromConverterList:**. Assume the Converter class has an instance method **converterName** that returns the name of the Converter object. Then the **objectAddedToConverterList:** method might have the following implementation:

```
- (void)objectAddedToConverterList:(NSNotification *)notification
{
    Converter *addedConverter = [notification object];

    // Add this to our popup (it will only be added if not there)...
    [myPopUpButton addItem:[addedConverter converterName]];
}
```

The Converters don’t need to know anything about the pop-up list or any other aspect of the user interface to your program.

If there are other objects of interest to the observer, place them in the notification’s optional dictionary or use **postNotificationName:object:userInfo:**.

Method Types

Accessing the default center	+ defaultCenter
Adding and removing observers	– addObserver:selector:name:object: – removeObserver: – removeObserver:name:object:
Posting notifications	– postNotification: – postNotificationName:object: – postNotificationName:object:userInfo:

Class Methods

defaultCenter

+ (NSNotificationCenter *)**defaultCenter**

Returns the current task’s notification center, which is used for system notifications.

nstance Methods

addObserver:selector:name:object:

– (void)**addObserver:**(id)*anObserver*
selector:(SEL)*aSelector*
name:(NSString *)*notificationName*
object:(id)*anObject*

Registers *anObserver* to receive notifications with the name *notificationName* and/or containing *anObject*. When a notification of name *notificationName* containing the object *anObject* is posted, *anObserver* receives an *aSelector* message with this notification as the argument. The method for the selector specified in *aSelector* must have one and only one argument. If *notificationName* is **nil**, the notification center notifies the observer of all notifications with an object matching *anObject*. If *anObject* is **nil**, the notification center notifies the object of all notifications with the name *notificationName*.

The notification center does not retain *anObserver* or *anObject*. Therefore, you should always send **removeObserver:** or **removeObserver:name:object:** to the notification center before releasing these objects.

postNotification:

– (void)**postNotification:**(NSNotification *)*notification*

Posts *notification* to the notification center. You can create *notification* with the NSNotification class method **notificationWithName:object:** or **notificationWithName:object:userInfo:**. An exception is raised if *notification* is **nil**.

See also: – **postNotificationName:object:**, – **postNotificationName:object:userInfo:**

postNotificationName:object:

– (void)**postNotificationName:**(NSString *)*notificationName*
object:(id)*anObject*

Creates a notification with the name *notificationName*, associates it with the object *anObject*, and posts it to the notification center. *anObject* is typically the object that is posting the notification. It may be **nil**.

This method invokes **postNotificationName:object:userInfo:** with a **userInfo:** argument of **nil**.

See also: – **postNotification:**

postNotificationName:object:userInfo:

– (void)**postNotificationName:**(NSString *)*notificationName*
object:(id)*anObject*
userInfo:(NSDictionary *)*userInfo*

Creates a notification with the name *notificationName*, associates it with the object *anObject* and dictionary *userInfo*, and posts it to the notification center. This method is the preferred method for posting notifications. *anObject* is typically the object that is posting the notification. It may be **nil**. *userInfo* also may be **nil**.

See also: – **postNotificationName:object:**

removeObserver:

– (void)**removeObserver:**(id)*anObserver*

Removes *anObserver* from all notification associations in the notification center. Be sure to invoke this method (or **removeObserver:name:object:**) before releasing *anObserver* or any object specified in **addObserver:selector:name:object:**.

removeObserver:name:object:

– (void)**removeObserver:**(id)*anObserver*
name:(NSString *)*notificationName*
object:(id)*anObject*

Removes *anObserver* as the observer of notifications with the name *notificationName* and object *anObject* from the notification center. Be sure to invoke this method (or **removeObserver:**) before deallocating the observer object or any object specified in **addObserver:selector:name:object:**.

If *anObserver* is **nil**, all objects are removed as observers of *notificationName* containing *anObject*. (Recall that the object that a notification contains is usually the object that posted the notification.) If *notificationName* is **nil**, *anObserver* is removed as an observer of all notifications containing *anObject*. If *anObject* is **nil**, *anObserver* is removed as an observer of *notificationName* containing any object. For example, if you wanted all objects to stop observing notifications containing the object **aWindow**, you would sent this message:

```
[[NSNotificationCenter defaultCenter] removeObserver:nil name:nil object:aWindow];
```