NSNotificationCenter

Inherits From: NSObject

Conforms To: NSObject (NSObject)

Declared In: Foundation/NSNotification.h

Class Description

An NSNotificationCenter object (or simply, *notification center*) is essentially a notification dispatch table. It notifies all observers of events meeting specific criteria of notification and sender. This event information is encapsulated in NSNotification objects, also known as *notification objects*, or simply*, notifications*. Client objects register themselves as observers of a specific notification originating in another object. When the condition occurs to signal a notification, some object (which may or may not be the object observed) posts an appropriate notification object to the notification center. (See the class specification of NSNotification for more on notification objects.) The notification center dispatches a message to each observer (using the selector provided by the observer), with the notification as the sole argument.

An object registers itself to observe notifications by the **addObserver:selector:name:object:** method, specifying the object and associated notification it wants to see. However, the observer need not specify both of these parameters. If it specifies only the object, it will see *all* notifications associated with that object. If the object specifies only a notification name to observe, it will see that notification for *any* object whenever it's posted.

The methods **postNotificationName:object:** and **postNotificationName:object:userInfo:** are provided as convenience methods, which both create and post notifications.

Each task has a default notification center.

As an example of using the notification center, suppose your program can perform a number of conversions on text (for instance, MIF to RTF or RTF to ASCII). You have defined a class of objects that perform those conversions, Convertor. Convertor objects might be added or removed during program execution. Your program has a client object that wants to be notified when convertors are added or removed, allowing the application to reflect the available options 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(objectAddedToConvertorList:)
```

```
        name:@"NSConverterAdded" object:nil];

    [[NSNotificationCenter defaultCenter] addObserver:self
        selector:@selector(objectRemovedFromConvertorList:
        name:@"NSConverterRemoved" object:nil];
```

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

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

or

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

The notification center identifies all observers who are interested in the ªNSConverterAddedº or ªNSConverterRemovedº 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 **objectAddedToConvertorList:** and **objectRemovedFromConvertorList:**. Assume the Convertor class has an instance method **convertorName** that returns the name of the Convertor object. Then the **objectAddedToConvertorList:** method might have the following implementation:

```
    - (void)objectAddedToConvertorList:(NSNotification *)notification
    {
        Convertor *addedConvertor = [notification object];

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

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

Accessing the Default Notification Center

+ (NSNotificationCenter *)**defaultCenter**

Returns the default notification center object; used for generic notifications.

Adding and Removing Observers

- (void)**addObserver:**(id)*anObserver*
selector:(SEL)*aSelector*
name:(NSString *)*aName*
object:(id)*anObject*

Registers *anObserver* and *aSelector* with the receiver so that *anObserver* receives an *aSelector* message when a notification of name *aName* is posted to the notification center by *anObject*. If *anObject* is **nil**, observer will get posted whatever the object is. If *aName* is **nil**, observer will get posted for all notifications that match *anObject*.

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

Removes *anObserver* as the observer of any notifications from any objects.

- (void)**removeObserver:**(id)*anObserver*
  name:(NSString *)*aName*
  object:*anObject*

Removes *anObserver* as the observer of *aName* notifications from *anObject*.

Posting Notifications

- (void)**postNotification:**(NSNotification *)*aNotification*

Posts *aNotification* to the notification center. Raises NSInvalidArgumentException if the name associated with *aNotification* is **nil**.

- (void)**postNotificationName:**(NSString *)*aName*
  object:(id)*anObject*

Creates a notification object that associates *aName* and *anObject* and posts it to the notification center.

- (void)**postNotificationName:**(NSString *)*aName*
  object:(id)*anObject*
  userInfo:(NSDictionary *)*userInfo*

Creates a notification object that associates *aName* and *anObject* and posts it to the notification center. *userInfo* is a dictionary of arbitrary data that will be passed with the notification. *userInfo* may be **nil**.