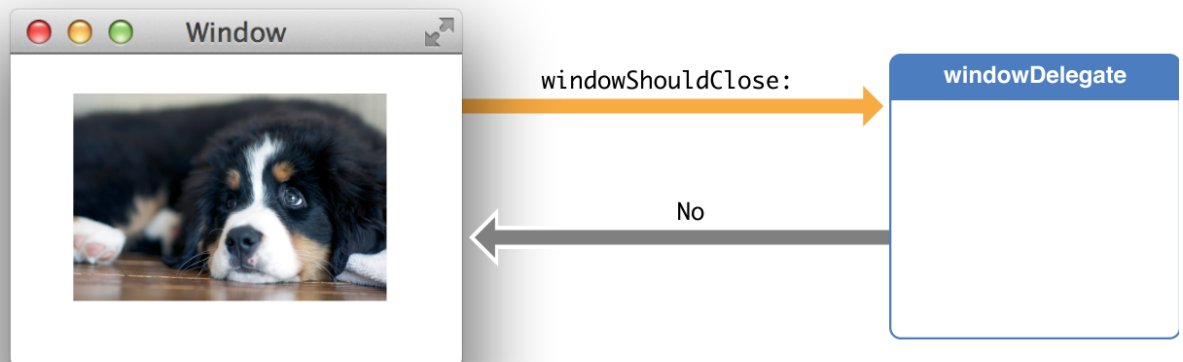# Delegation

Delegation is a simple and powerful pattern in which one object in a program acts on behalf of, or in coordination with, another object. The delegating object keeps a reference to the other object—the delegate—and at the appropriate time sends a message to it. The message informs the delegate of an event that the delegating object is about to handle or has just handled. The delegate may respond to the message by updating the appearance or state of itself or other objects in the application, and in some cases it can return a value that affects how an impending event is handled. The main value of delegation is that it allows you to easily customize the behavior of several objects in one central object.

Delegation and the Cocoa Frameworks

The delegating object is typically a framework object, and the delegate is typically a custom controller object. In a managed memory environment, the delegating object maintains a weak reference to its delegate; in a garbage-collected environment, the receiver maintains a strong reference to its delegate. Examples of delegation abound in the Foundation, UIKit, AppKit, and other Cocoa and Cocoa Touch frameworks.

An example of a delegating object is an instance of the `NSWindow` class of the AppKit framework. `NSWindow` declares a protocol, among whose methods is `windowShouldClose:`. When a user clicks the close box in a window, the window object sends `windowShouldClose:` to its delegate to ask it to confirm the closure of the window. The delegate returns a Boolean value, thereby controlling the behavior of the window object.

Delegation and Notifications

The delegate of most Cocoa framework classes is automatically registered as an observer of notifications posted by the delegating object. The delegate need only implement a notification method declared by the framework class to receive a particular notification message. Following the example above, a window object posts an `NSWindowWillCloseNotification` to observers but sends a `windowShouldClose:` message to its delegate.

Data Source

A data source is almost identical to a delegate. The difference is in the relationship with the delegating object. Instead of being delegated control of the user interface, a data source is delegated control of data. The delegating object, typically a view object such as a table view, holds a reference to its data source and occasionally asks it for the data it should display. A data source, like a delegate, must adopt a protocol and implement at minimum the required methods of that protocol. Data sources are responsible for managing the memory of the model objects they give to the delegating view.

Prerequisite Articles

Class definition

Related Articles

Notification

Protocol

Controller object

Definitive Discussion

Delegates and Data Sources

Sample Code Projects

UITableView Fundamentals for iOS