15

Navigation Controllers

A navigation controller is a class that manages the presentation of a stack of view controllers one at a time. The topmost item on the stack is visible, and users can navigate down the stack one view controller at a time. Whenever a view controller is pushed on—or off the navigation controller's stack—iOS applies an appropriate slide animation automatically. Navigation controllers are implemented in the `UINavigationController` class in the `UIKit` framework and can be found in several standard applications such as the iOS Mail, and Settings apps.

ADDING A NAVIGATION CONTROLLER TO A STORYBOARD

To create a navigation controller using the interface editor, simply select the storyboard scene that you want to use as the root view controller of the navigation stack and select Editor ➪ Embed In ➪ Navigation Controller. You can optionally drag a Navigation Controller object from the Object library to the storyboard. When you create a navigation controller in this manner, Xcode creates a default scene that is set up to act as the root view controller for the navigation controller (see Figure 15-1).

In most cases, you will want to use one of the existing scenes in the storyboard as the root view controller. To do this, first select the Relationship Segue between the navigation controller and the default root view controller and delete it (see Figure 15-2).

Now select the navigation controller scene, hold down the Ctrl key, and drag from the navigation controller scene to whatever scene you want to use as the root view controller. When you release the mouse pointer you will be presented with a list of segue types to use; select Relationship Segue (see Figure 15-3).

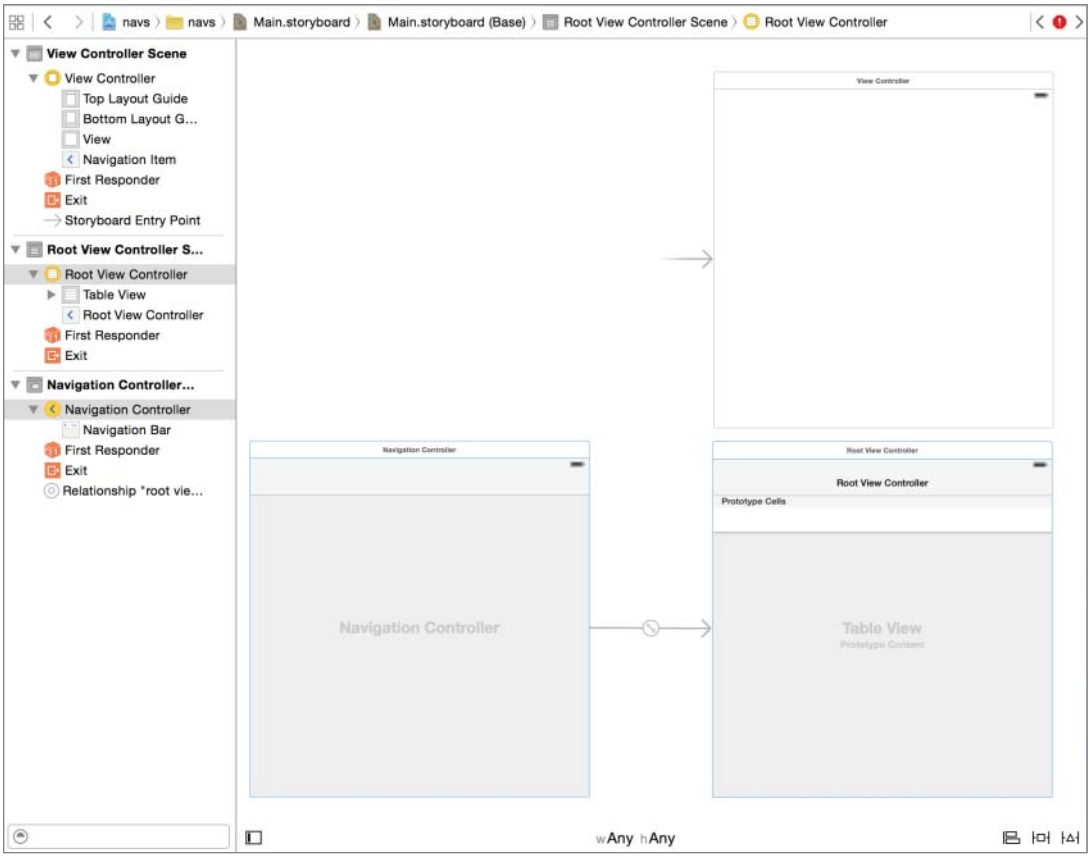

FIGURE 15-1

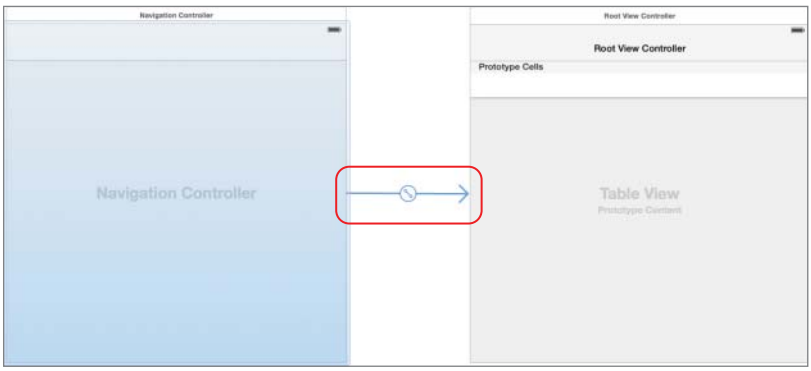

FIGURE 15-2

You can now delete the previous root view controller scene, which is now unused if you wish. If the navigation controller is going to be the primary view controller of your application, then you must

ensure that the Is Initial View Controller option in the Attribute Editor is selected for the navigation controller (see Figure 15-4).

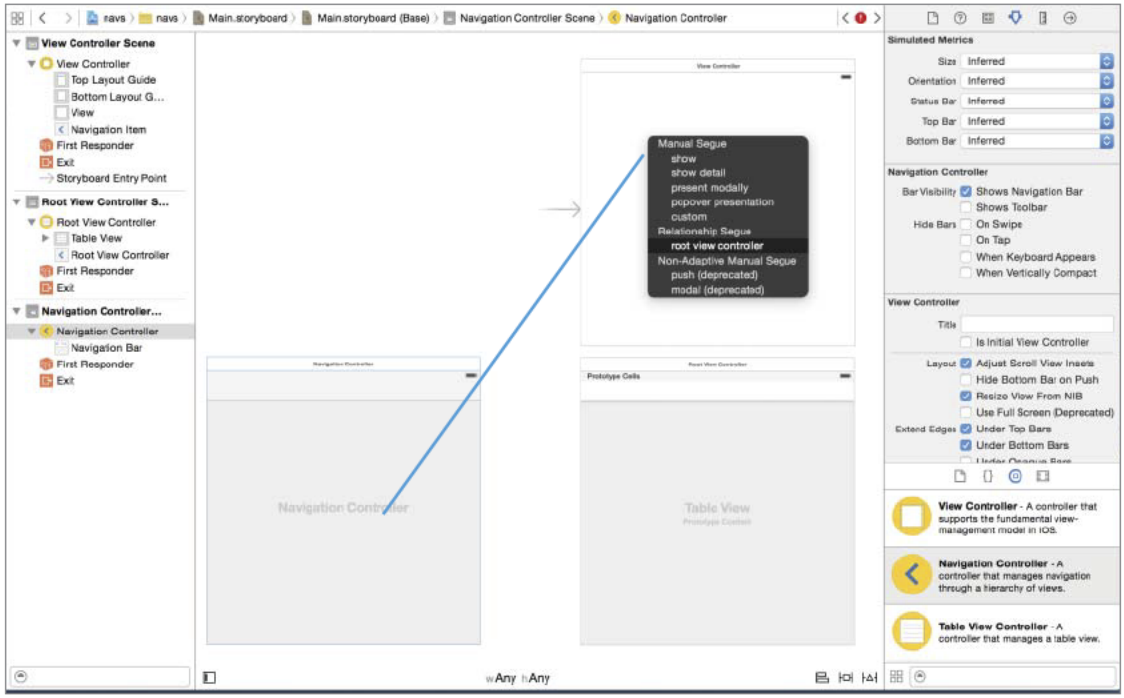

FIGURE 15-3

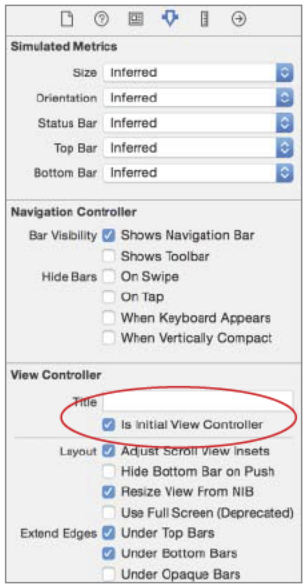

FIGURE 15-4

THE NAVIGATION CONTROLLER INTERFACE

A navigation controller contains two key components, as shown in Figure 15-1.

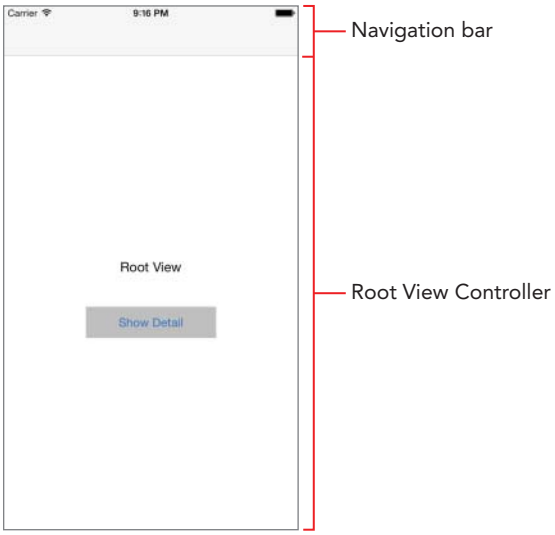

FIGURE 15-5

- **Navigation bar:** This is the horizontal header on the top of the view, just below the status bar; it typically contains the title of the view being displayed and an optional back button.
- **Root view controller:** This is the base view controller at the bottom of the navigation stack; it cannot be removed from the navigation controller. When this view controller is visible, there is no back button available to the user.

To set the title that is displayed in the navigation bar when a view controller is on the top of the stack, you can set up the view controller's title property as follows:

```
self.title = "Root View";
```

You can add buttons to the navigation bar that perform custom actions. The following code snippet adds a Share button to the right side of the navigation bar (see Figure 15-6). When this button is tapped, the `onShare` method will be called.

```
override func viewDidLoad()
{
    super.viewDidLoad()

    self.title = "Root View";

    let shareButton:UIBarButtonItem = UIBarButtonItem(barButtonSystemItem:
        UIBarButtonSystemItem.Action,
        target: self,
        action: "onShare:")
```

```
        self.navigationItem.setRightBarButtonItem(shareButton, animated: false)
    }

    func onShare(sender: UIBarButtonItem) {

    }
```

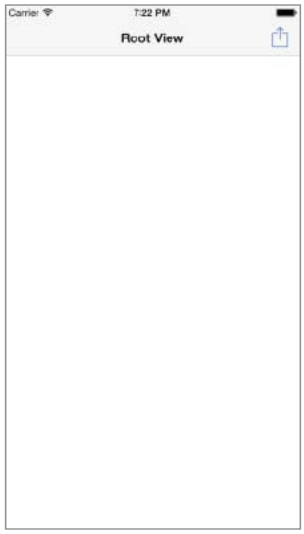

FIGURE 15-6

You can add and remove view controllers onto the navigation stack by using the following methods:

```
pushViewController(viewController, animated)
popViewControllerAnimated(animated)
```

The `UINavigationController` class provides the following two additional methods that enable you to pop all view controllers down to a specific view controller:

```
popToRootViewControllerAnimated(animated: Bool)
popToViewController(viewController, animated)
```

TRY IT

In this Try It, you create a new Xcode project based on the Single View Application template called `NavigationControllerTest` that uses a navigation controller to manage a hierarchy of views.

> **REFERENCE** *The code for this Try It is available at* `www.wrox.com/go/swiftios`.

Lesson Requirements

- Launch Xcode.
- Create a new project based on the Single View Application template.
- Edit the storyboard with Interface Builder.
- Embed the default storyboard scene in a navigation controller.
- Add a button to the default scene.
- Add a second scene to the storyboard.
- Create a segue from the button in the first scene to the second scene.

Hints

- To show the Object library, select View ➪ Utilities ➪ Show Object Library.
- To show the assistant editor, select View ➪ Assistant Editor Show Assistant Editor.

Step-by-Step

- Create a Single View Application in Xcode called `NavigationControllerTest`.
  1. Launch Xcode and create a new application by selecting File ➪ New ➪ Project.
  2. Select the Single View Application template from the list of iOS project templates.
  3. In the project options screen, use the following values:
     - **Product Name:** NavigationControllerTest
     - **Organization Name:** your company
     - **Organization Identifier:** com.yourcompany
     - **Language:** Swift
     - **Devices:** iPhone
     - **Use Core Data:** Unchecked
     - **Include Unit Tests:** Unchecked
     - **Include UI Tests:** Unchecked
  4. Save the project onto your hard disk.
- Add a `UILabel` instance to the default scene.
  1. From the Object library, drag and drop a Label object onto the scene and position it beneath the picker.
  2. Edit the text displayed in the label to Root View.

3. Select the label in the scene and click the Align button to display the alignment constraint editor. Add a constraint to center the label horizontally.
4. Select the label in the scene and click the Align button to display the alignment constraint editor. Add a constraint to center the label vertically.
5. Select the label in the scene and select Editor ➪ Size to Fit Contents to ensure the label is large enough to show its contents.
6. Update the frames to match the constraints you have set.
    - Click on the View controller item in the dock above the storyboard scene. This is the first of the three icons located directly above the selected storyboard scene.
    - Select Editor ➪ Resolve Auto Layout Issues ➪ Update Frames.

- Add a button to the storyboard.
    1. From the Object library, select a button and drop it onto the scene.
    2. Double-click the button and change the text displayed in it to `Show Detail`.
    3. Drag the button to position it near the center of the scene, beneath the label. The precise size or position does not matter.
    4. Use the Attribute inspector to change the background color of the button to a shade of gray. The background color attribute is located in the View subsection of the Attribute inspector; you may need to scroll down a little to access it.
    5. Ensure the button is selected; if it is not, simply click it once.
    6. Center the button horizontally by selecting Editor ➪ Align ➪ Horizontal Center in Container.
    7. Ensure the button is selected and use the Pin button to display the constraints editor popup.
        - Pin the width of the button to 165.
        - Pin the height of the button to 40.
        - Pin the distance between the button and the label to 50.
        - Click the Add 3 Constraints button to dismiss the constraints editor popup.
    8. Update the frames to match the constraints you have set.
        - Click on the View controller item in the dock above the storyboard scene. This is the first of the three icons located directly above the selected storyboard scene.
        - Select Editor ➪ Resolve Auto Layout Issues ➪ Update Frames.
- Embed the default scene in a navigation controller.

1. Click on the View controller item in the dock above the storyboard scene. This is the first of the three icons located directly above the selected storyboard scene.
2. Select Editor ➪ Embed In ➪ Navigation Controller to embed the default scene as the root view controller of a navigation controller. Your storyboard should resemble Figure 15-7.

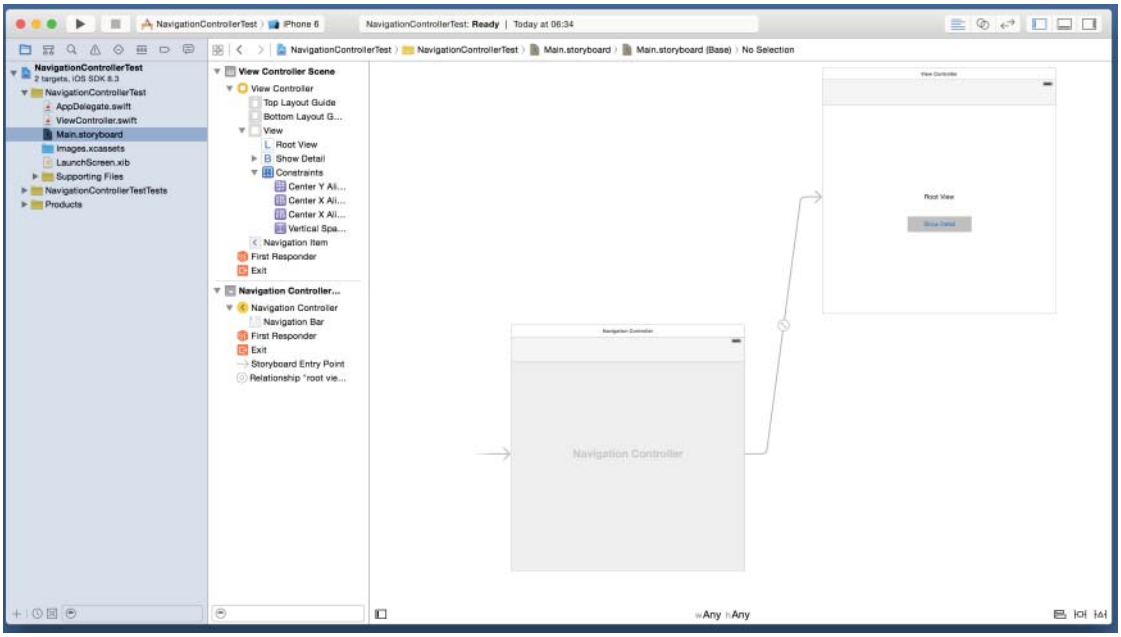

FIGURE 15-7

➤ Add a new subclass of `UIViewController` called `DetailViewController`.

1. Ensure the project navigator is visible.
2. Right-click the `NavigationControllerTest` group and select New File from the context menu.
3. Select the Cocoa Touch Class template and click Next.
4. Call the new class `DetailViewController` and ensure that the new class is a subclass of `UIViewController` by selecting `UIViewController` in the drop-down combo box.
5. Ensure that the Also create XIB file option is unchecked and click Next.
6. Select a folder where files should be created. It is best to accept the default location provided by Xcode.

➤ Create a new scene in the storyboard.

1. Ensure the `Main.storyboard` file is open. If it is not, then select it in the project navigator.
2. Drag a View Controller object from the Object library onto the storyboard canvas.

3. Double-click the canvas to zoom out.
4. Position the new scene alongside the original scene.
5. Select the new scene in the storyboard, select the View Controller object from the dock, and use the Identity inspector to change its Custom Class to `DetailViewController`. To show the Identity inspector, select View ➪ Utilities ➪ Show Identity inspector.

- Add a `UILabel` instance to the new scene.

1. From the Object library, drag and drop a Label object onto the scene and position it beneath the picker.
2. Edit the text displayed in the label to Detail View.
3. Select the label in the scene and click the Align button to display the alignment constraint editor. Add a constraint to center the label horizontally.
4. Select the label in the scene and click the Align button to display the alignment constraint editor. Add a constraint to center the label vertically.
5. Select the label in the scene and choose Editor ➪ Size to Fit Contents to ensure the label is large enough to show its contents.
6. Update the frames to match the constraints you have set.
    - Click on the View controller item in the dock above the storyboard scene. This is the first of the three icons located directly above the selected storyboard scene.
    - Select Editor ➪ Resolve Auto Layout Issues ➪ Update Frames.

- Create a segue from the button in the first scene to the new scene.

1. Double-click the canvas to zoom out. Position the two scenes sufficiently apart on the canvas by dragging them.
2. Double-click the first scene to activate it.
3. Right-click the Show detail button in the first scene to bring up a context menu. Drag from the circle beside the action item under the `Triggered Segues` category in the context menu to the second scene.
4. When you release the mouse button, you will be asked to select the segue type. Select `Show`.

- Test your app in the iOS Simulator.

1. Click the Run button in the Xcode toolbar. Alternatively, you can select Project ➪ Run.
2. Tap on the Show Detail button and observe the second scene pushed onto the navigation controller stack.

> **REFERENCE** *To see some of the examples from this lesson, watch the Lesson 15 video online at* `www.wrox.com/go/swiftiosvid`.