Storyboards Tutorial

Based on Chapter 4 (pages 116 - 173) from iOS 5 by Tutorials; refer to that chapter for a more detailed explanation of the steps involved in this tutorial.

Getting Started: Start a new project in XCode.

-- **Template:** Single View Application

-- **Product Name:** Ratings

-- **Devices:** iPhone only

-- **Use Automatic Reference Counting:** Checked

-- **Use Storyboards:** Checked

Part 1

Objective: Set up the application to use storyboards and add a tab view controller.

1. Select the project in the Project Explorer, and in the **Summary** tab, deselect the *Landscape Left* and *Landscape Right* orientations. This will make sure the application runs only in the *Portrait* orientation.

2. Open *MainStoryboard.storyboard.* Show the **File Inspector** (View-->Utilities-->Show File Inspector). This will open the panel on the right hand side, if it's not already open. Scroll down to find the **Use Autolayout** checkbox and deselect this checkbox.

3. Open the **Object Library** and drag a **Tab Bar Controller** onto the canvas. Note: It may be necessary to zoom out to see the new view controllers that were added to the scene.

4. Drag a label onto the view controller labelled *Item 1* and set the text of that label to be "First Tab". Drag a label onto the view controller labelled *Item 2* and set the text of that label to be "Second Tab".

> **Note:** You can't drag stuff into the scenes when the editor is zoomed out. You'll need to return to the normal zoom level first. You can do that quickly by double-clicking in the canvas.

5. Select the Tab Bar Controller and go to the Attributes inspector. Check the box that says Is Initial View Controller.

   What this does: This means that when you run the app, UIApplication will make

the Tab Bar Controller the main screen. The storyboard always has a single view controller that is designated the *initial view controller*, that serves as the entry point into the storyboard.

6. **RUN THE APP.** The app now has a tab bar and you can switch between the two view controllers with the tabs.

7. Remove the view controller that was originally added by the template, as you'll no longer be using it. The storyboard now contains just the tab bar and the two scenes for its tabs.

Part 2 (page 129)

Objective: Add a table view controller.

8. Click on the first view controller (the child of the Tab Bar Controller) to select it, and then delete it. From the Object Library drag a new Table View Controller into the canvas in the place where that previous scene used to be.

9. With the Table View Controller selected, choose **Editor\Embed In\Navigation Controller** from Xcode's menubar. This adds yet another view controller to the canvas. (You could also have dragged in a Navigation Controller from the Object Library, but this Embed In command is just as easy.)

10. Let's connect these two new scenes to the Tab Bar Controller. Ctrl-drag from the Tab Bar Controller to the Navigation Controller. Choose the **Relationship Segue – view controllers** option. This creates a new relationship arrow between the two scenes.

11. Select the **Tab Bar Controller**, and notice that there is a new tab called "Item"; this represents the link to the new navigation view controller. Drag this item to the left of "Item 2" so that it is the first tab.

> **Note:** If Interface Builder doesn't let you drag to rearrange the tabs inside the Tab Bar Controller, then temporarily go to another file in the project and switch back to the storyboard again. That seems to fix it. Or follow the iOS developer's motto: When in doubt, restart Xcode.

12. RUN THE APP. The first tab now contains a table view inside a navigation controller.

13. Select the Tab Bar Item inside the Navigation Controller, and in the Attributes inspector set its Title to "Players".

14. Rename the Tab Bar Item for the view controller from the second tab to "Gestures".

15. The resources for this chapter contain a subfolder named Images. Add that folder to the project. Then, in the Attributes inspector for the Players Tab Bar Item, choose the **Players.png** image. You probably guessed it, give the Gestures item the image **Gestures.png**.

16. Select the Navigation Item for the Table View Controller and change its title in the Attributes inspector to "Players".

17. **RUN THE APP.** Marvel at your pretty tab bar, all without writing a single line of code!

Part 3 (page 135)

Objective: Add 'prototype cells' to the project so the table view can show some custom content.

18. In the **Table View Controller**, select the **Table View Cell**. It is a child of the **Table View**. Open the **Attributes Inspector** and change the **Style** of the table view cell from *Custom* to *Subtitle*. This immediately changes the appearance of the cell to include two labels.

19. With the **Table View Cell** still selected, and the **Attributes Inspector** still open, set the Accessory attribute to Disclosure Indicator and give the cell the Reuse Identifier **PlayerCell**.

20. Add a new file to the project. Choose the **Objective-C class** template. Name the class **PlayersViewController** and make it a subclass of **UITableViewController**. The **With XIB for user interface** option should be unchecked because you already have the design of this view controller in the storyboard. No nibs today!

21. Go back to the storyboard and select the Table View Controller. In the Identity inspector, set its Class to **PlayersViewController**. That is the essential step for hooking up a scene from the storyboard with your own view controller subclass. Don't forget this or your class won't be used!

22. Add a mutable array property to **PlayersViewController.h**.

```
#import <UIKit/UIKit.h>

@interface PlayersViewController : UITableViewController

@property (nonatomic, strong) NSMutableArray *players;

@end
```

23. Add a new file to the project using the **Objective-C class** template. Name it **Player**, subclass of **NSObject**.

24. Add the following properties to **Player.h**.

```
@property (nonatomic, copy) NSString *name;
@property (nonatomic, copy) NSString *game;
@property (nonatomic, assign) int rating;
```

25. Import **Player** and **PlayersViewController** into **AppDelegate.m**.

26. Add a new instance variable to **AppDelegate** called *_players* that is an NSMutableArray datatype. Note the underscore.

    HINT: NSMutableArray *_players;

27. Add the following code to the **didFinishLaunchingWithOptions:** selector of **AppDelegate.m.** Do this just above the last line of the existing code.

```
_players = [NSMutableArray arrayWithCapacity:20];
Player *player = [[Player alloc] init];
player.name = @"Bill Evans";
player.game = @"Tic-Tac-Toe";
player.rating = 4;
[_players addObject:player];

player = [[Player alloc] init];
player.name = @"Oscar Peterson";
player.game = @"Spin the Bottle";
player.rating = 5;

[_players addObject:player];
player = [[Player alloc] init];
player.name = @"Dave Brubeck";
player.game = @"Texas Hold'em Poker";
player.rating = 2;

[_players addObject:player];
```

WHAT THIS DOES: This simply creates some `Player` objects and adds them to the `_players` array.

28. Continue to edit the **didFinishLaunchingWithOptions** selector by adding the following code.

```
UITabBarController *tabBarController = (UITabBarController
*)self.window.rootViewController;

UINavigationController *navigationController = [tabBarController
viewControllers][0];

PlayersViewController *playersViewController = [navigationController
viewControllers][0];
```

WHAT THIS DOES: This creates references to all of the viewControllers that were created in the storyboard. The *Initial View Controller* property of the Tab Bar Controller that was set made that the **rootViewController** for the application window. The other two view controllers are referenced as children of that view controller.

29. Lastly, add the following line of code to **didFinishLaunchingWithOptions**.

```
playersViewController.players = _players;
```

WHAT THIS DOES: This sets the *players* property of the *playersViewController* to the value of the *_players* array set in the appDelegate.

Part 4 (page 140)

Objective: Set up the *datasource* for the table view so it can display the data.

30. Import the *Player* class into *PlayersViewController.*

31. In *PlayersViewController.m* change the return value of the *numberOfSectionsInTableView:* selector so that it returns a value of 1.

NOTE: There are a few new selectors that have been added to this class because it subclassed a **UITableViewController** when it was created. The template for

that class added these new selectors.

32. In *PlayersViewController.m* change the return value of the numberOfRowsInSection: selector so that it returns the count of the *players* array.

    HINT: [self.players count];

33. Modify the *cellForRowAtIndexPath:* selector to include the following code.

```
UITableViewCell *cell = [tableView
dequeueReusableCellWithIdentifier:@"PlayerCell"];
Player *player = (self.players)[indexPath.row];
cell.textLabel.text = player.name;
cell.detailTextLabel.text = player.game;
return cell;
```

    WHAT THIS DOES: It creates a cell object and sets it's properties to those of the *player* object.

    NOTE: Running the application at this point will cause an error. Why? (The answer is in the next step!)

34. Select the mainStoryboard.storyboard file, select the **Table View Cell,** and set the **Identifier** property to "PlayerCell", the same value as was used above.

35. RUN THE APP. The list of names should appear in the table cells.