**iPhone Development Tutorial II**
**January 2011**

**Manish Chaturvedi**

## Contents

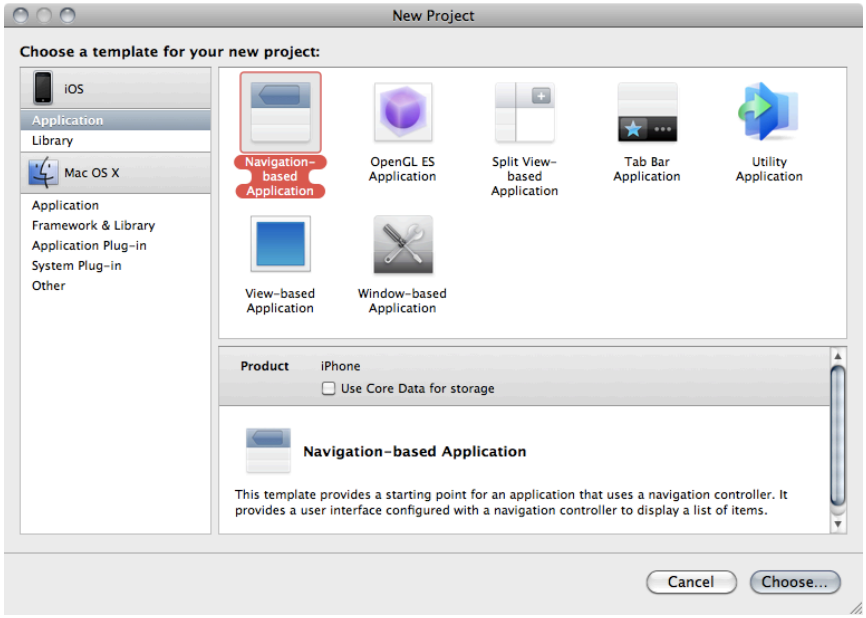
The following tutorial introduces creating a **Navigation Based Project** and handling the **Table View** towards developing iOS SDK based applications using Objective C.

A Navigation Based application in conjunction with the Table View widget is best suited to display Hierarchical Data. Each successive layer of detail resides on it's own dedicated view. Navigation View Controllers provide the means to move forward and back through the screens. Table View presents data in form of rows of data. Example of such an application is the Apple Email App.

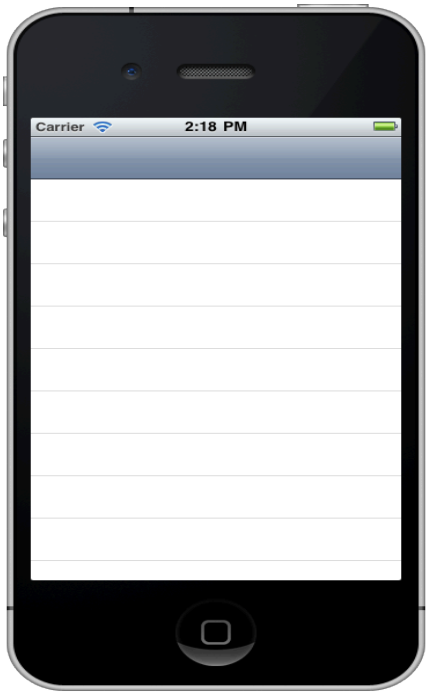
## Create Navigation Based Application

➔ **"File > New Project >iOS>Application>Navigation Based Application**

The newly created Project should be named keeping in mind that auto generated application objects (Classes, Xib files etc.) Are named accordingly.

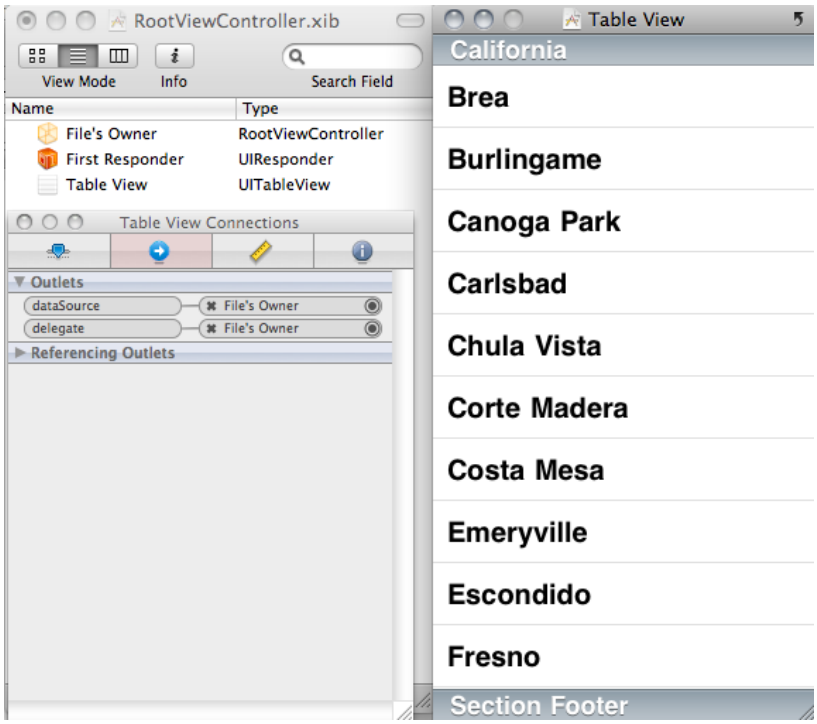



**Build and Run** the newly created project.




*Default Navigation Based Application in the Simulator*

**Explore RootViewController in Interface Builder**




**Modify RootViewController.m**

We need to display some text on the Main View when application is launched. Edit `cellForRowAtIndexPath` method and set the text for each Cell

```
// Customize the appearance of table view cells.
- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath {


// Configure the cell.
      cell.textLabel.text = @"NFL";
```

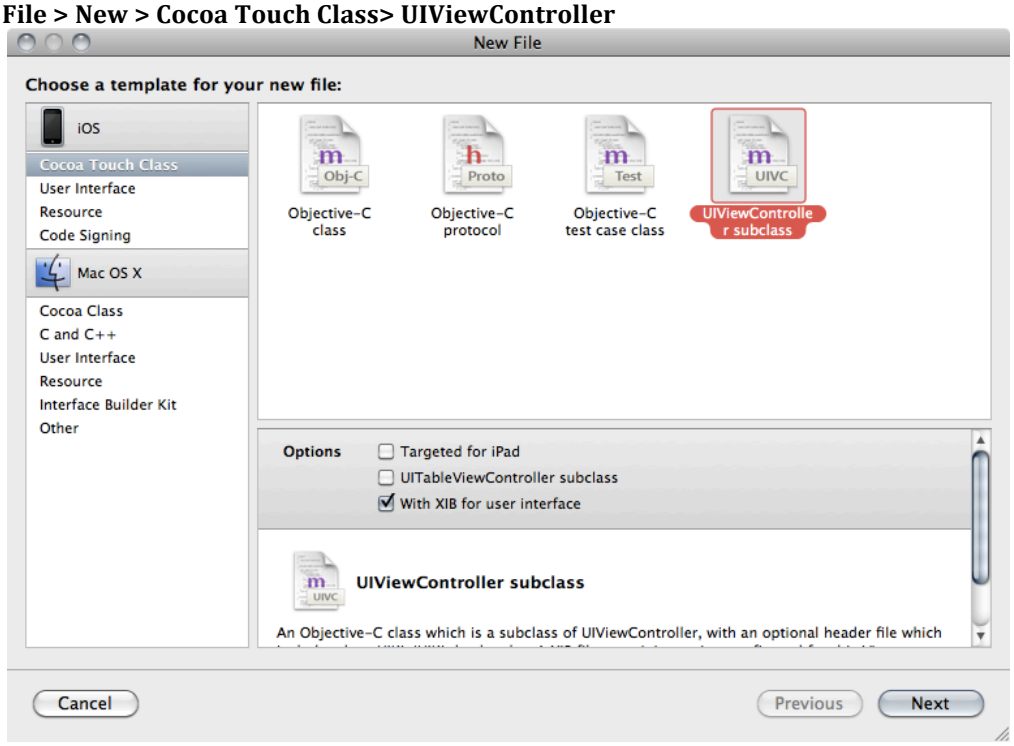
Next Specify Number of Rows for the Main View

```
// Customize the number of rows in the table view.
- (NSInteger)tableView:(UITableView *)tableView
numberOfRowsInSection:(NSInteger)section {
    return 5;
```

Now we are ready to define a Custom view to display next level of data.

**Create New View Controller**

**File > New > Cocoa Touch Class> UIViewController**

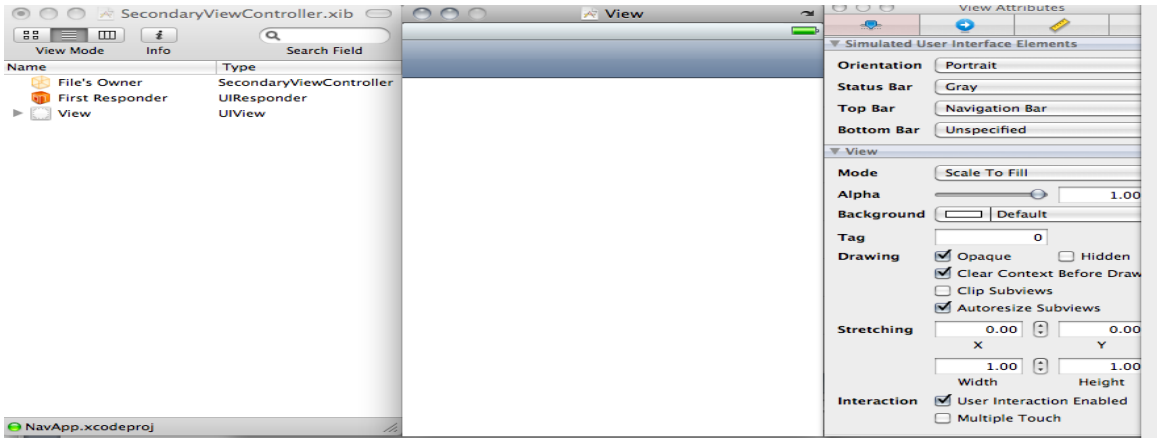



Name the newly created View Controller **SecondaryViewController**

A xib file is also created. Move it to the *Resources* Folder.

**Modify SecondaryViewController in Interface Builder**

- Set Navigation Bar at Top of Secondary View Controller in the Interface Builder.
- Add a Label to the View as well.



**Modify View Controller Header File**

- Add IBOutet for the Label to `SecondaryViewController.h`
- Call it **message**.
- Set Property and Synthesize attributes for the UILabel **message**.

**Modify RootViewController Implementation File**

- Add import statement for the newly created ViewController header to RootViewController.m
- Modify didSelectRowAtIndexPath method

Uncomment the provided code.
Use SecondaryViewController in place of the default:-

```
SecondaryViewController *detailViewController = [[….
```

- Make sure the correct Nib file name is provided for initWithNibName property.

Build and Run to observe the transition from Main View to the Secondary

- Next, Set title to the Main View by modifying **viewDidLoad** method in RootViewController.m

```
self.title = @"NFL Teams"
```

Build and Run, Note navigating back from Secondary View is now possible.

- Add a NSString *QtrBack to the SecondaryViewController.h

Remember to set the Property and Synthesize directives for this text string, which will be used on the Secondary View.

QtrBack can be set from the RootViewController.m, when a row is selected.

- Modify `didSelectRowAtIndexPath method`

```
SecondaryViewController *secondaryVC = [[SecondaryViewController alloc]
initWithNibName:@"SecondaryViewController" bundle:nil];
   secondaryVC.QtrBack = @"NFL QB";
```

- Uncomment and Modify `viewDidLoad` method in SecondaryViewController.h
  To set the Label text with the value in QtrBack string and Title of the view:-

```
message.text = QtrBack;
self.title = @"Team Details";
```

`Build and Run`

The Label Text does not appear – Fix the issue in Interface Builder.

**Exploring Table Views**

Table Views have a Single Column Data Display. Each Row has a UITableViewCell object.

- Data Source
  - How Many Rows?
  - What Does Each Row Contain?
- Delegation
  - What Happens When Row is Selected?

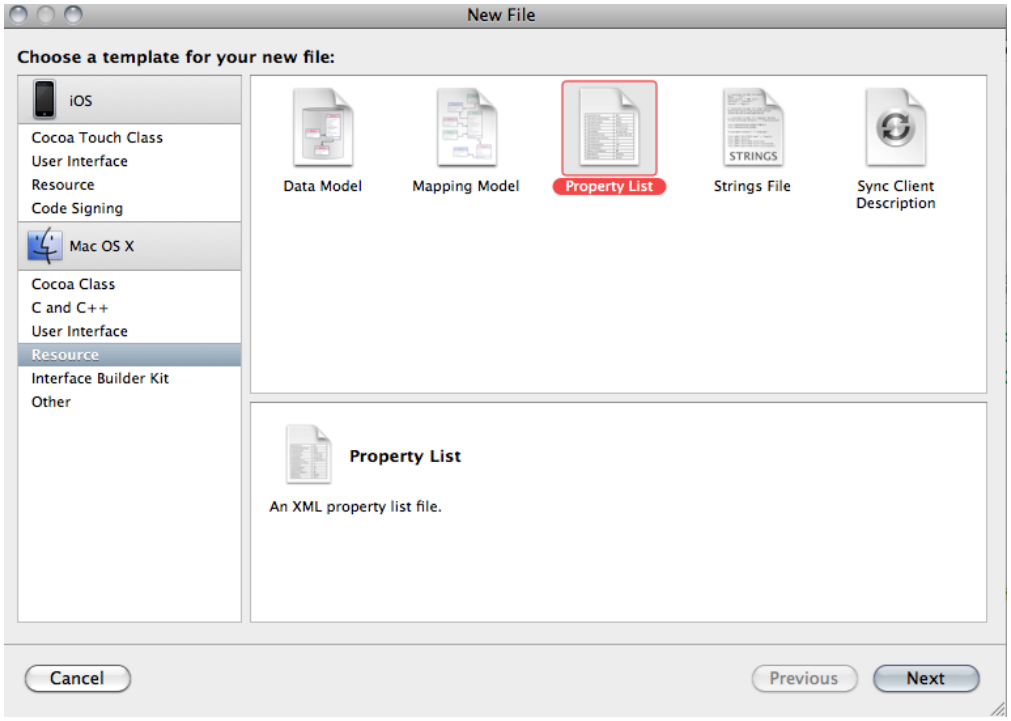
Typically Data Source and Delegate are contained within the View Controller.

- Open RootViewController.xib, Selecting the Table View.
- Bring up "Connections Inspector".

Confirm **datasource** and **delegate** outlets are mapped to the File's Owner.

**Creating Data for Table View**

We can hard code the Table View data in the Controller class, in form of an Array. A bit more flexible option is to create an external file – a **Property List** file




- Create NFLRoster.plist under the Resources Folder in XCode.
- Edit the plist by entering an Array of **Five** Item Values for different NFL Team names.

**Edit RootViewController.h**

Ensure that RootViewController handles the Protocol for the Data Source and Delegate outlets setup from the IB.

- Modify the header file.

```
@interface RootViewController : UITableViewController
<UITableViewDelegate, UITableViewDataSource>
```

- Create a **NSArray** object - Call it **teams**

**Edit RootViewController.m**

Next we move to the RootViewController implementation file.

- Edit **viewDidLoad** method to populate **teams** array with data from the Property List NFLRoster.

- Access the plist file from the Project resources:-

```
NSString *filePtr = [[NSBundle mainBundle]
                                    pathForResource:@"NFLRoster"
ofType:@"plist"];
```

- Populate the array **teams:-**
- Allocate storage for the array, and initialize with File.

```
[[NSArray alloc] initWithContentsOfFile:file
```

Next we need to implement **required** methods from the UITableDataSource protocols.

Documentation for the desired interface can be accessed by **Alt-DoubleClick** on the interface name, in this case <UITableViewDataSource>

UITableDataSource Required Methods:-

- tableView:numberOfRowsInSection:
- tableView:cellForRowAtIndexPath:

- Find **No Of Rows In Section** by counting # rows in teams array:-

```
return teams.count;
```

- Setting the cell For Each Row involves 3 distinct steps
  - ➔ Create the Cell (Memory allocation, Cell Style, Reuse Definition)

```
UITableViewCell *cell = [[UITableViewCell alloc]

initWithStyle:UITableViewCellStyleDefault
                                    reuseIdentifier:@"cell"];
```

  - ➔ Fill with Content

```
cell.textLabel.text = [teams objectAtIndex:indexPath.row];
```

  - ➔ Return Created cell

**Exercise**

- Set QtrBack value based on the Team Row clicked on the Main View.
- Modify the UI Interface Orientation to Landscape
- Modify cellForRowAtIndexPath to introduce a cell accessoryType
- Set it to aUITableViewCellAccessoryDetailDisclosure value.

## Expected OutCome