Tutorial - Part II
Data driven app using UINavigationController and UITableViewController

Step 9 - Continued from Part 1

1. Open the app that you completed in Part 1. Alternatively, open the app as it is found in the folder "DataBasedApp-Part2Beginning". This is your starting point.

Step 2 - Show a new page when clicking on a UITableCellView

2. Create a new class file that is a UIViewController subclass and call it *CityInformationViewController*. In the second screen of the wizard, make sure that this class subclasses *UIViewController*, and *With XIB for user interface* is checked.
3. Open the file *EMTableViewController.m* and look for the selector **didSelectRowAtIndexPath**.
4. Modify the code that is commented out there so that it looks like the following.

   ```
   CityInformationViewController *cityInfoViewController =[[CityInformationViewController alloc] initWithNibName:@"CityInformationViewController" bundle:nil];
   [self.navigationController pushViewController:cityInfoViewController animated:YES];
   ```

   What does this code do? This will create a new view controller, and push it onto the navigationController. Pushing a viewController onto a navigationController causes the view to be visible. The *animated* parameter will determine if it will slide in from the right hand side or not.

RUN: If you run the app at this point, and select one of the table cells, you should now see a new view controller appear. Selecting back will take you back to the table.

Step 3 - Move the data to a model.

5. Right now the data (the list of city names) is found in the file *EMTableViewController*. This makes it difficult to share with another viewController. Let's make a new file and move the array to that new file.
6. Import the DataModel class (both the .h and .m) into your project by selecting "File-->Add Files to...". Navigate to the DataModel files in the "DataBasedApp-Part2Beginning" folder and select them. **Be sure to have "Copy items into destination group's folder (if needed)" selected.**
7. Open and take a look at those 2 files in XCode. You should see a property that has been synthensized called *cities*. You should also see the array that has been created that holds all of the city names.
8. Now let's remove the existing array from the tableViewController. We need to remove it and replace it with a reference to the DataModel.
9. In *EMTableViewController.h* replace the **tableContent** property with a new property like this:

```
@property (nonatomic, strong) DataModel *data;
```

10. Be sure to import the *DataModel.h* file into *EMTableViewController.h*.
11. Be sure to synthesize the **data** property in *EMTableViewController.m.*
12. Now let's create the actual data object. In the *initWithStyle* selector, under the comment that says "Custom initialization", add the following line of code.

```
self.data = [DataModel getInstance];
```

13. Now we can make a reference to the data object.
14. In the *numberOfRowsInSection* selector, modify the value that is being returned to be the following. Notice how it is simply making a reference to the data object.

```
return [self.data.cities count];
```

15. Lastly, let's modify the text that is being shown. In the *cellForRowAtIndexPath* selector, modify the line the sets the cell.textLabel.text to the be the following:

```
[self.data.cities objectAtIndex:indexPath.row];
```

Step 4 - Remembering the selection

16. Let's add a property to the *DataModel* that is an NSInteger that will keep track of what UITableViewCell was clicked on. The code for *DataModel.h* looks like this:

```
@property (nonatomic) NSInteger selectedCity;
```

Don't forget to synthesize in the .m file!

17. In the *didSelectRowAtIndexPath* at *EMTableViewController.m,* just before the viewController is created and pushed onto the navigationController, set the selectedCity like so:

```
self.data.selectedCity = indexPath.row;
```

Step 5 - Remembering the selection

18. In the CityInformationViewController class, create a property for the **data** object in the exact same way it was created in the *EMTableViewController* class. (hint: import, @property, @synthesize).
19. In the *viewDidLoad* selector of the *CityInformationViewController* class, add the following lines of code:

```
self.data = [DataModel getInstance];
```

```
self.title = [self.data.cities objectAtIndex:self.data.selectedCity];
```

RUN: Now when you run the app, and click on the city in the table, a new viewController will appear that will have the selected city name at the top of the navigationController.

Step 6 - Expanding the data.

20. Let's expand the data so that it includes more than just a city name.
21. Add the *City* class to your project (it's in the completed folder). Notice all of the properties: name, province, thumbnail, imageURL.
22. Now we'll change our data model to use the new properties.
23. Import "City.h" into the *DataModel.h* file. We can't use it if we don't import it!
24. Modify the code in the *init* selector to replace the existing array with the following array:

```
City *toronto = [[City alloc] init];
toronto.name = @"Toronto";
toronto.province = @"Ontario";
toronto.thumbnail = @"toronto_thumb.png";
toronto.imageURL = @"http://www.eberhardt.ca/ocad/toronto.png";

self.cities = [NSArray arrayWithObjects:toronto, nil];
```

NOTE: We are adding a City object to the array, not an NSString.

25. In the *cellForRowAtIndexPath* selector of EMTableViewController, change the code that modified the cell.textLabel.text to the following:

```
cell.textLabel.text = [(City*)[self.data.cities objectAtIndex:indexPath.row] name];
```

NOTE: We are accessing the name property of the city object, not the object in the array itself.

26. Let's change the reference in the *CityInformationViewController.m* file.

```
self.title = [(City*)[self.data.cities objectAtIndex:self.data.selectedCity] name];
```

27. Add a few more cities - you'll find images for Vancouver and Quebec in the *beginning folder.*

Step 7 - Showing an image of the city.

28. Add a UIImageView object to the *CityInformationViewController.xib* file. Be sure it is a child of the main view that is found in that XIB file.
29. Create a referencing outlet for that new UIImageView to the File's Owner and call it *cityImage.*
30. Let's use the *imageURL* property of our city object to download an image from the web. In the *viewDidLoad* selector of *CityInformationViewController.m* file, add the following code:

```
self.cityImage.image = [UIImage imageWithData:[NSData dataWithContentsOfURL:[NSURL
URLWithString:[(City*)[self.data.cities objectAtIndex:self.data.selectedCity] imageURL]]]];
```

31. This long line of code does the following (working backwards):
    a. accesses the imageURL property of the selectedCity that was stored in the data object.
    b. converts the NSString (imageURL) to an NSURL object.
    c. creates an NSData object with the result of the NSURL.
    d. creates an image from the NSData object.

RUN: Running the app, you should now see an image when going to that page.