Navigating a Data Hierarchy with Table Views

Hierarchical Data Models and Table Views

> The Data Model as a Hierarchy of Model Objects

1. You can describe model objects in terms of their properties. These properties are of two general kinds: attributes and relationships.

2. Attributes represent elements of model-object data.

3. It is through these relationships that a data model acquires hierarchical depth by composing an object graph.

> Table Views and the Data Model

1. Arrays contain strings or other elements that a table view can use when displaying a row content.

2. In many of the methods defined for a table view's data source and delegate, the table view passes in an index path to identify the row that is the focus of the current operation. An index path is an instance of the Foundation framework's NSIndexPath class.

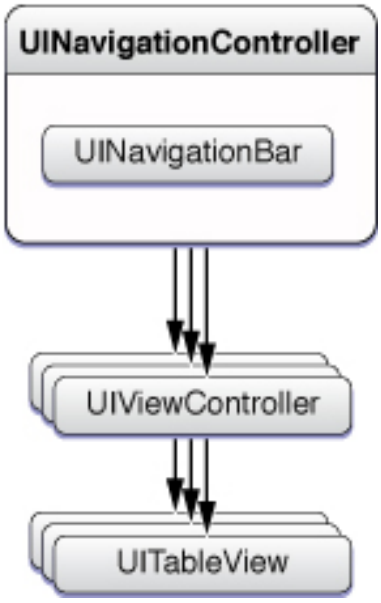
3. Mapping levels of the data model to table views

View Controllers and Navigation-Based Apps

> Navigation Controllers

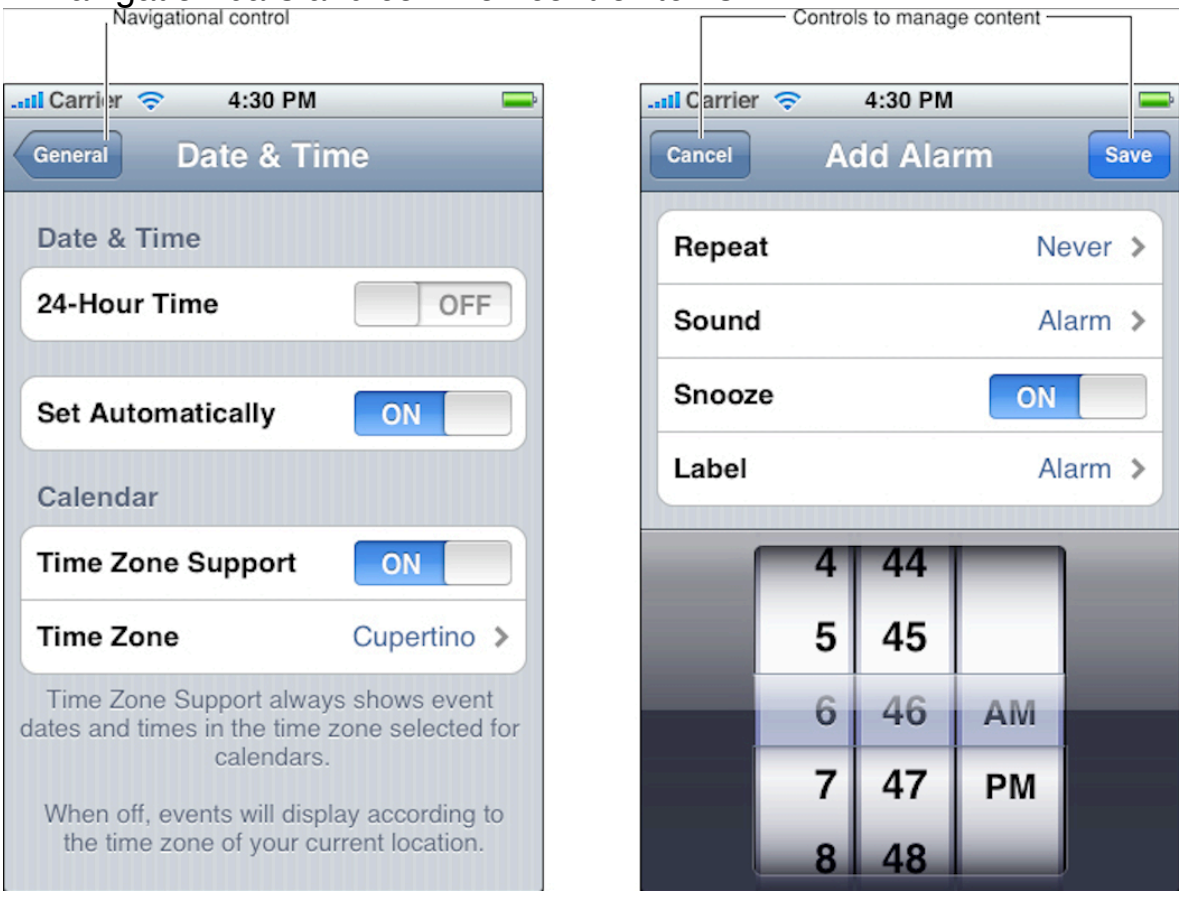
1. When the user tap a row of the table view, the root view controller pushes the next view controller onto the stack. The new view controller's table view visually slides into space from the right. When users tap back button in the navigation bar, the current view controller is popped off the stack.

Navigation controller and view controllers in a navigation-based app



> Navigation Bars

1. Navigation bars and common control items



2. A UINavigationController manages the navigation bar, including the items displayed in the bar. A UIVieController object manages a view displayed below the navigation bar.

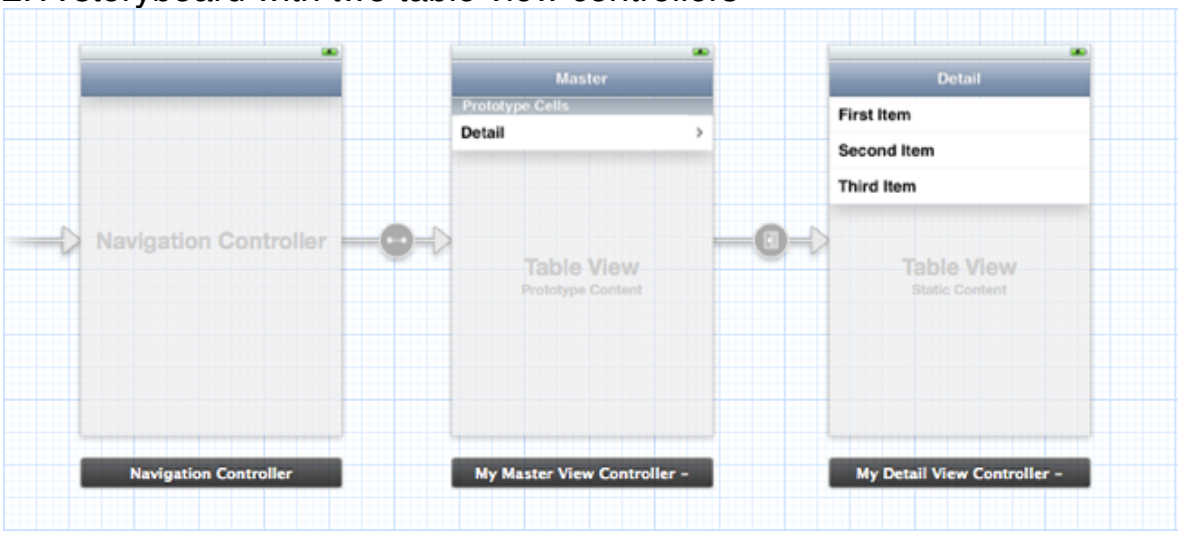
> Table View Controllers
1. The table view controller sets itself as the data source and the delegate of the table view.

2. When the table view is about to appear for the first time, the table view controller sends reloadData to the table view.

3. The UITableViewController class implements the foregoing behavior by overriding loadView, viewWillAppear:, and other methods inherited from UIViewController.

> Managing Table Views in a Navigation-Based App
1. When a user taps a row of the table view, the table view calls the tableView:didSelectRowAtIndexPath: or tableView:accessoryButtonTappedForRowWithIndexPath: method implemented by delegate. The delegate creates the table view controller managing the next table view in sequence.

2. A storyboard with two table view controllers



=> A **scene** represents an onscreen content area that is managed by a view controller.
=> A **relationship** is a type of connection between scenes.
=> A **segue** represents a transition from one scene to the next scene.

3. Passing data to a destination view controller

```
- (void)prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender
{
  if ([[segue identifier] isEqualToString:@"ShowDetails"]) {
    MyDetailViewController *detailViewController = [segue
```

```
destinationViewController];
    NSIndexPath *indexPath = [self.tableView
indexPathForSelectedRow];
    detailViewController.data = [self.dataController
objectInListAtIndex:indexPath.row];
  }
}
```

4. You create a delegate protocol that declares methods that the destination view controller calls when it needs to pass back some data.
Passing data back to a source view controller

```
@protocol MyAddViewControllerDelegate <NSObject>
- (void)addViewControllerDidCancel:(MyAddViewController *)controller;
- (void)addViewControllerDidFinish:(MyAddViewController *)controller
data:(NSString *)item;
@end

- (void)addViewControllerDidCancel:(MyAddViewController *)controller {
  [self dismissViewControllerAnimated:YES completion:NULL];
}

- (void)addViewControllerDidFinish:(MyAddViewController *)controller
data:(NSString *)item {
  if ([item length]) {
    [self.dataController addData:item];
    [[self tableView] reloadData];
  }
  [self dismissViewControllerAnimated:YES completion:NULL];
}
```

Design Pattern for Navigation-Based Apps
A navigation-based app with table views should follow these design best practices:
=> A view controller (typically a subclass of UITableViewController), acting in the role of data source, populates its table view with data from an object representing a level of the data hierarchy.

=> The view controller stores the data it needs for populating its table view.

=> The current view controller on top of the navigation-controller stack

creates the next view controller in the sequence and, before it pushes it onto the stack, sets the data that this view controller, acting as data source, needs to populate its table view.