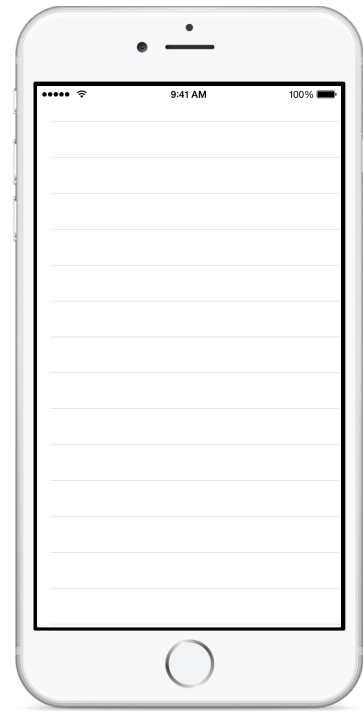# Journal
Lesson 3

Description

Add a table view controller class to the project, and bind it to the table view controller in the storyboard.

Learning Outcomes

- Practice binding user-defined classes to view controllers with Interface Builder.

- Discover the datasource and delegate dependencies of table views.

- Discover how table view controllers automatically assign a table view delegate and datasource.

- Describe the concepts of delegates and delegation, and relate them to protocols.

- Discover how table views efficiently manage memory occupied by table cells.

Vocabulary

table view controller	table view	UITableViewController
UITableView	delegate	delegation
protocol	UITableViewDataSource	UITableViewDelegate
table cell prototype	cell prototype identifier	

Materials

- **Journal Lesson 3** Xcode project
- **Table View Controllers** presentation

- **Delegates and Delegation** presentation
- **Protocols** presentation

Opening

How can we get some rows of data to appear in the table view?

Agenda

- Discuss how a table view controller manages a collection of cells, and manages the user interaction of scrolling and selecting table rows.
- Using Interface Builder, select the table view controller, open the Identity Inspector (⌥⌘3), and observe how the **Class** managing the table is a default `UITableViewController`.
- Add a new Cocoa Touch Class (⌘N) to the project called JournalTableViewController and specify a **Subclass of** `UITableViewController`.

  ```
  import UIKit

  class JournalTableViewController: UITableViewController {
  ...
  ```

- Discuss that, by specifying a `UITableViewController` subclass, Xcode generates a class definition with commonly used boilerplate code.
- Using Interface Builder, select the table view controller, and use the Identity Inspector (⌥⌘3) to change the **Class** attribute to `JournalTableViewController`.
- Run the app (⌘R), observe the empty table rows, and draw attention to the warning raised by Xcode.
- Discuss the "prototype cells must have reuse identifier" warning raised by Xcode.
- Using Interface Builder and the Document Outline (▢), expand the hierarchy of the Journal Table View Controller, and observe how the Table View contains a Table View Cell and Content View.
- Using Interface Builder and the Document Outline (▢), Control-click the Table View and observe how the `JournalTableViewController` is configured to act as the `delegate` and `dataSource` for the table view.
- Present the concept of table view controllers.
- Using the Documentation and API Reference (⇧⌘0), explore the `UITableViewController` class reference, and observe how it conforms to the `UITableViewDelegate` and the `UITableViewDataSource` protocols.
- Present the concept of protocols.

- Using the Documentation and API Reference (⇧⌘0), explore the `UITableViewDelegate` and the `UITableViewDataSource` protocol references.

- Present the concept of delegates and delegation.

- Using Interface Builder, select the prototype cell in the table view controller, open the Attributes Inspector (⌥⌘4), set the **Identifier** attribute to **JournalEntryCell**, and observe the Xcode warning disappear.

- Discuss how the value **JournalEntryCell** will serve as the cell reuse identifier when the table view displays the cell for a table row.

- Add a `cellIdentifier` property to the `JournalTableViewController` class.

```
class JournalTableViewController: UITableViewController {

    let cellReuseIdentifier = "JournalEntryCell"
    ...
```

- Using the Xcode Documentation and API Reference (⇧⌘0), explore the `UITableViewDelegate` protocol reference, drawing attention to the `numberOfSectionsInTableView:`, `tableView:numberOfRowsInSection:`, and `tableView:cellForRowAtIndexPath:` methods.

- Update the implementation of the `numberOfSectionsInTableView:` method.

```
override func numberOfSectionsInTableView(tableView: UITableView) ->
    Int {
    return 1
}
```

- Explain how table views in some apps may consist of multiple sections, but the journal entry list will only consist of a single section.

- Discuss how the value returned by `tableView:numberOfRowsInSection:` will eventually be determined by the data model.

```
override func tableView(tableView: UITableView,
    numberOfRowsInSection section: Int) -> Int {
    // TODO: determined by number of journal entries
    return 0
}
```

- Uncomment and update the boilerplate `tableView:cellForRowAtIndexPath:` method implementation.

```
override func tableView(tableView: UITableView, cellForRowAtIndexPath
    indexPath: NSIndexPath) -> UITableViewCell {
    let cell =
        tableView.dequeueReusableCellWithIdentifier(cellReuseIdentifier,
            forIndexPath: indexPath)
    // Configure the cell...
    return cell
}
```

- Discuss how the `cellReuseIdentifier` property value matches the cell prototype **Identifier** attribute specified in Interface Builder.
- Run the app (⌘R), observe the empty table cells appear, and observe that Xcode does not report any warnings.

Closing

What is the significance of the `#MARK` comments we see in the `JournalTableViewController` implementation?

Modifications and Extensions

- Create one new class that adopts the `UITableViewDataSource` protocol, and another class that adopts the `UITableViewDelegate` protocol. Delete the existing `dataSource` and `delegate` connections of the table view, and re-establish the connections to instances of the delegate and protocol classes you create.

Resources

Table View Programming Guide for iOS https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/TableView_iPhone/AboutTableViewsiPhone/AboutTableViewsiPhone.html

Cocoa Core Competencies: Delegation https://developer.apple.com/library/ios/documentation/General/Conceptual/DevPedia-CocoaCore/Delegation.html

Cocoa Core Competencies: Protocol https://developer.apple.com/library/ios/documentation/General/Conceptual/DevPedia-CocoaCore/Protocol.html

UIKit User Interface Catalog: Table Views https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/UIKitUICatalog/UITableView.html

UITableViewController Class Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UITableViewController_Class/index.html

UITableView Class Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UITableView_Class/index.html

UITableViewDelegate Protocol Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UITableViewDelegate_Protocol/index.html

UITableViewDataSource Protocol Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UITableViewDataSource_Protocol/index.html