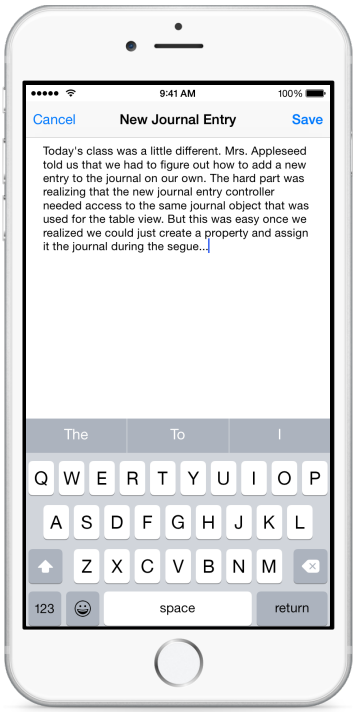
Journal
Lesson 11

Description

Implement the behavior of a save button, adding a new journal entry to the journal.



Learning Outcomes

- Practice adding interface elements to a view, and establishing an action connection between a view and controller.
- Describe how segue identifiers may be used to distinguish one segue from another.
- Describe how to pass data between controllers during a segue transition.
- Discover how to update the rows displayed in a table view when data changes.
- Assess an existing app implementation and plan additional features.

Vocabulary

navigation bar	bar button item	controller action
segue	segue identifier	`UIStoryboardSegue`
`UIViewController`	`UITableView`	

Materials

- **Journal Lesson 11** Xcode project

Opening

How can we add the new journal entry to the journal?

Agenda

- Discuss how a save button could be added to the `NewJournalViewController` interface; and how the controller needs to create a new `JournalEntry` and add it to the `Journal`.
- Using Interface Builder and the Object Library (⌥⌘L), add a new Bar Button Item to the right side of the New Journal Entry View Controller navigation bar., and use the Attributes Inspector (⌥⌘4) to change the Identifier attribute to **Save**.
- Using the Assistant Editor (⌥⌘↩), Control-drag from the Save button to the `NewJournalEntryViewController` class implementation to create a new action called `save:`.

```
@IBAction func save(sender: UIBarButtonItem) {
}
```

- Discuss how the `NewJournalEntryViewController` will use the content of the text view to create a new `JournalEntry` object.
- Using the Assistant Editor (⌥⌘↩), add an outlet property by Control-dragging from the text view to the `NewJournalEntryViewController` class implementation.

```
@IBOutlet weak var journalEntryContents: UITextView!
```

- Discuss how the `NewJournalEntryViewController` will need access to the `Journal` object owned by the `JournalTableViewController`, in order to add a new `JournalEntry` to the `Journal`.
- Add a `Journal` property to the `NewJournalEntryViewController` class.

```
var journal: Journal?
```

- Discuss that the property is an optional type, since the controller initializer will not assign a value to the property.
- Implement the `save:` method.

```
@IBAction func save(sender: UIBarButtonItem) {
   let entry = JournalEntry(date: NSDate(),
      contents: journalEntryContents.text)
   journal?.addEntry(entry)
   dismissViewControllerAnimated(true, completion: nil)
}
```

- Discuss how the `save:` method creates a new `JournalEntry` object with the text entered in the text view, uses optional chaining to add the JournalEntry to the Journal, and then dismisses the view controller.
- Discuss how the `NewJournalEntryViewController` might obtain access to the same `Journal` object maintained by the `JournalTableViewController`.
- Using Interface Builder, select the segue indicator between the Journal Table View Controller and the New Journal Entry View Controller, and use the Attributes Inspector (⌥⌘4) to set the Identifier attribute to `newJournalEntry`.
- Update the `JournalTableViewController` by adding a property for the new segue identifier and updating the method `prepareForSegue:sender:`.

```
let newJournalEntrySegueIdentifier = "newJournalEntry"
...
override func prepareForSegue(segue: UIStoryboardSegue, sender: AnyObject?) {
    if segue.identifier == journalEntrySegueIdentifier {
        ...
    } else if segue.identifier == newJournalEntrySegueIdentifier {
        if let newJournalEntryViewController =
            segue.destinationViewController as? NewJournalEntryViewController {
            newJournalEntryViewController.journal = journal
        }
    }
}
```

- Discuss how the `prepareForSegue:sender:` method will assign the `Journal` object to the `NewJournalEntryViewController` before the segue completes.
- In the `JournalTableViewController`, reduce the number of sample journal entries to make it easier to see that a new `JournalEntry` appears in the table view.

```
...
let journal = Journal(entries: (0..<3).map {
    JournalEntry(date: NSDate(), contents: "Contents for entry \($0)")
})
...
```

- Run the app (⌘R), tap the compose button, enter some text, tap the Save button, and witness that the table view reappears but the new `JournalEntry` does not appear in the table.
- Discuss how the table view itself is not aware that a new `JournalEntry` has been added to the `Journal`.
- Using the Documentation and API Reference (⇧⌘0), explore the `UITableView reloadData` method.
- In the `JournalTableViewController`, add an implementation of `viewWillAppear:`.

```
override func viewWillAppear(animated: Bool) {
    super.viewWillAppear(animated)
    tableView.reloadData()
}
```

- Discuss that, every time the view will appear, the `reloadData` method will be called.
- Run the app (⌘R), add a new journal entry, tap the save button, and observe that a new row for the journal entry appears in the table view.

Closing

What happens if we create a bunch of journal entries, and then quit and restart the app? How might we save the journal, and load it when the app starts? How would you implement a feature that allows the user to edit existing journal entries?

Modifications and Extensions

- Investigate how to serialize an array to a file on an iOS device, and ensure the journal is saved appropriately. Load the journal data when the app starts.
- Investigate how to save and load the journal data with iCloud or Core Data.
- Present a different view to the user when the journal is empty, prompting them to create a new entry.
- Investigate how to modify the app icon with a badge representing the number of entries in the journal.
- Allow the user to select their favorite journal entries, and integrate a tab bar controller that displays another table view that presents the favorite journal entries.

Resources

Xcode Overview: Build a User Interface https://developer.apple.com/library/ios/documentation/ToolsLanguages/Conceptual/Xcode_Overview/edit_user_interface.html

View Controller Programming Guide for iOS: Presenting View Controllers https://developer.apple.com/library/ios/featuredarticles/ViewControllerPGforiPhoneOS/ModalViewControllers/ModalViewControllers.html

The Swift Programming Language: Optional Chaining https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/OptionalChaining.html

UIViewController Class Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIViewController_Class/index.html

UIStoryboardSegue Class Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIStoryboardSegue_Class/index.html

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