Journal Lesson 5

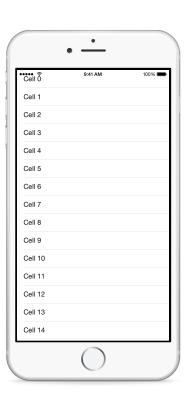


Description

Add a JournalEntry model to the project.

Learning Outcomes

- Practice analyzing model requirements and implementing a class.
- Practice declaring stored properties and computed properties.
- Describe the purpose of initialization.
- Apply an NSDateFormatter to customize the string representation of a date and time.
- Describe how protocols describe behavior to which class implementations can conform.
- Apply the Printable protocol to control the string representation of an object.



Vocabulary

model	property	default property value
NSDate	NSDateFormatter	initializer
protocol	protocol adoption	computed property

Materials

- · Journal Lesson 5 Xcode project
- · Initialization presentation

Opening

What model do we need in our app?

Agenda

- Discuss the need for a JournalEntry model to represent a journal entry with a date and contents.
- Add a new (%N) JournalEntry class to the project.

```
import Foundation

class JournalEntry {
    let date: NSDate
    let contents: String
    let dateFormatter = NSDateFormatter()
}
```

- Discuss how the dateFormatter property relies on a default value, but that the date and contents properties will require an initializer.
- Using the Documentation and API Reference (☆ ※ 0), explore the NSDate and NSDateFormatter class references.
- Add an initializer to the JournalEntry class.

```
init(date: NSDate, contents: String) {
    self.date = date
    self.contents = contents
}
```

- Present the concept of initialization.
- Discuss how, when a JournalEntry is instantiated, the default NSDateFormatter value is assigned, the initializer prepares both the date and contents properties, and configures the dateFormatter.
- Experiment with the JournalEntry model by printing a JournalEntry object in viewDidLoad.

```
override func viewDidLoad {
    super.viewDidLoad()
    let entry = JournalEntry(date: NSDate(), contents: "A happy day!")
    println("Entry: \((entry)\)")
}
```

- Run the app (***R**), and observe the console output (**☆ *C**) display name of the app module followed by the type, Journal Journal Entry.
- Discuss the goal of seeing the date and time of the JournalEntry when using string interpolation with the object itself.
- Using the Documentation and API Reference (♠ **%0**), explore the Printable protocol reference.
- Update the JournalEntry class with a Printable protocol adoption and a description computed property.

```
class JournalEntry: Printable {
    ...
    var description: String {
        return dateFormatter.stringFromDate(date)
    }
}
```

- Explain how adopting the Printable protocol and conforming with a description property enables customization of the textual representation of an object.
- Run the app (***R**), and observe the console output (**☆ *C**) display the formatted date and time of the JournalEntry object.
- Remove the JournalEntry instantiation and the println call from viewDidLoad.

```
override func viewDidLoad() {
   super.viewDidLoad()
}
```

Closing

How many NSDateFormatter objects would be created in our app if the JournalEntry description computed property instantiated an NSDateFormatter rather than using a default property value for the dateFormatter property?

Modifications and Extensions

• Enhance the implementation of the JournalEntry description method to include a truncated portion of its contents, or **(Empty)** if the contents string is empty.

Resources

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

The Swift Programming Language: Classes and Structures https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/ClassesAndStructures.html

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

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

Swift Standard Library Reference: Printable https://developer.apple.com/library//ios/documentation/General/Reference/SwiftStandardLibraryReference/Printable.html

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

NSDate Class Reference https://developer.apple.com/library/ios/documentation/Cocoa/Reference/Foundation/Classes/NSDate Class/

NSDateFormatter Class Reference https://developer.apple.com/library/ios/documentation/Cocoa/Reference/Foundation/Classes/NSDateFormatter Class/