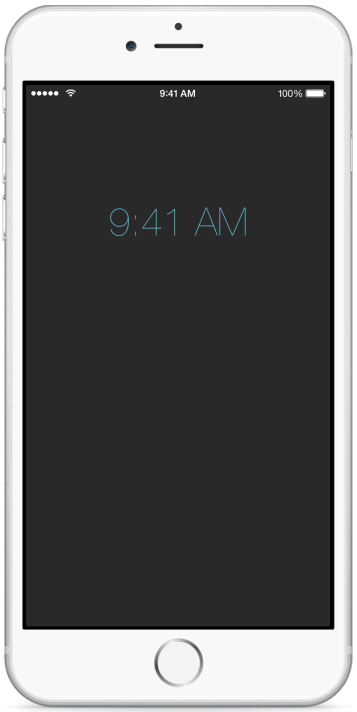
Clock
Lesson 2

Description

Add a `Clock` model and introduce MVC. Display the current time when the app starts.



Learning Outcomes

- Classify the components of MVC and assess how the model, view and controller manifest in an iOS app.
- Write a Swift class definition with methods and simple computed properties.
- Practice instantiating objects and applying string interpolation.
- Describe the Swift property declaration syntax with a specified default value.
- Apply `NSDate` and `NSDateFormatter` to create a formatted string for display.

Vocabulary

MVC design pattern	model	class
encapsulation	object	instantiation
method definition	return type	`NSDate`
computed property	`return`	initializer
property default value	`NSDateFormatter`	string interpolation
enumeration		

Materials

- **Clock Lesson 2** Xcode project
- **Model-View-Controller** presentation

Opening

How do we display the current time on the screen? Where is the clock in our code?

Agenda

- Present the concept of the MVC pattern, and how models, views and controllers manifest in an iOS app.
- Discuss the need for a model to encapsulate the representation of a clock.
- Add a `Clock` class to the project (⌘N).
- Using the Xcode Documentation and API Reference (⇧⌘0), explore the `NSDate` class.
- Define and implement a `currentTime` method.

```
func currentTime() -> NSDate {
   return NSDate()
}
```

- Explain the anatomy and syntax of a Swift method definition, emphasizing the return type, name, camelCase convention, and body.
- Discuss how the `currentTime` method always returns a new instance of an `NSDate` object.
- Explain how Swift provides a feature known as "computed properties" that represent properties whose values are computed each time they are accessed.
- Replace the `currentTime` method definition with a computed property.

```
var currentTime: NSDate {
   return NSDate()
}
```

- Explain the computed property syntax, type annotation, and implicit `get` implementation.
- Declare a `clock` property within the ViewController class.

```
let clock = Clock()
```

- Explain the anatomy of the property declaration and its syntax.
- Briefly explain the approach of declaring the `clock` property default value, which is assigned during initialization.
- Update `viewDidLoad` to set the label text with the raw `NSDate` object returned by the `Clock` `currentTime` property.

```
override func viewDidLoad() {
   super.viewDidLoad()
   timeLabel.text = "\(clock.currentTime)"
}
```

- Run the app (⌘R) to witness the need to customize the format of the `NSDate` as a string.
- Using the Xcode Documentation and API Reference (⇧⌘0), explore the `NSDateFormatter` class and the `NSDateFormatterStyle` constants.
- Use an `NSDateFormatter` to display a properly formatted time on the screen.

```
func viewDidLoad() {
   super.viewDidLoad()
   let formatter = NSDateFormatter()
   formatter.timeStyle = .ShortStyle
   timeLabel.text = formatter.stringFromDate(clock.currentTime)
}
```

- Explain how the shorthand enumeration syntax can be used when its type can be inferred.
- Run the app (⌘R), and witness the correctly formatted time on the screen.

Closing

What happens if we send the app to the background and then restore it later? Why?

Modifications and Extensions

- Modify the `Clock` class to change the `currentTime` method to a readable and writable computed property.
- Change the `clock` property to a `var` and remove the default value. Investigate why the property must have an explicit datatype, and why it must be an optional type.

Resources

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

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

Searching Developer Documentation http://developer.apple.com/library/ios/recipes/xcode_help-documentation_organizer/SearchingDocumentation/SearchingDocumentation.html

Start Developing iOS Apps Today: Finding Information https://developer.apple.com/library/ios/referencelibrary/GettingStarted/RoadMapiOS/FindingInformation.html

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

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

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

The Swift Programming Language: Optionals https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/TheBasics.html#//apple_ref/doc/uid/TP40014097-CH5-ID330

The Swift Programming Language: String Interpolation https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/StringsAndCharacters.html#//apple_ref/doc/uid/TP40014097-CH7-ID292

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

The Swift Programming Language: Enumeration Syntax https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/Enumerations.html#//apple_ref/doc/uid/TP40014097-CH12-ID146