

Clock

Lesson 5

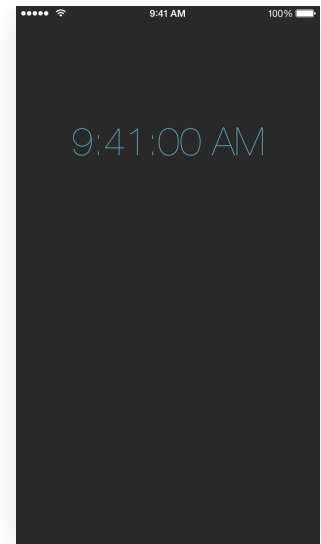


Description

Use `NSTimer` to execute a method repeatedly, to keep the currently displayed time correct. Deprecate the use of notifications and observers for this feature.

Learning Outcomes

- Define what a feature requirement is.
- Assimilate user expectations and feature requirements.
- Interpret what threads and run loops are, and relate them to `NSTimer`.
- Use `NSTimer` to execute a method repeatedly without blocking the main run loop.
- Apply forced unwrapping to unwrap an optional property value.



Vocabulary

| | | |
|---------------------|-----------------|----------------------|
| feature requirement | user experience | run loop |
| thread | blocking | <code>NSTimer</code> |
| class method | optional | optional binding |
| timer invalidation | | |

Materials

- **Clock Lesson 5** Xcode project
- **Run Loops and NSTimer** presentation

Opening

How often do we want the displayed time to update, and how might we continuously update the displayed time?

Agenda

- Discuss how a person would use the Clock app and what the experience should be. Discuss the main flaw in the app: time is only updated when bringing the app into the foreground, and the displayed time does not continuously change while the app is running.
- Present the concept of run loops, threads and the `NSTimer` class.
- Add a new controller property for an `NSTimer`.

```
var timer: NSTimer?
```

- Discuss how the timer property is declared as an optional, because the `ViewController` initializer will not initialize the property.
- Using the Xcode Documentation and API Reference (⌘⌘0), explore the `NSTimer` class reference and its `scheduledTimerWithTimeInterval:target:selector:userInfo:repeats:` class method.
- Replace the observer registration in `viewDidLoad` with the creation of an `NSTimer` that will call `updateTimeLabel` every second.

```
override func viewDidLoad() {  
    super.viewDidLoad()  
    timer = NSTimer.scheduledTimerWithTimeInterval(1.0, target: self,  
        selector: "updateTimeLabel", userInfo: nil, repeats: true)  
}
```

- Discuss the meaning of the arguments passed to `scheduledTimerWithTimeInterval:target:selector:userInfo:repeats:`.
- Modify the `updateTimeLabel` method's format of the displayed time, such that it displays seconds.

```
func updateTimeLabel() {  
    let formatter = NSDateFormatter()  
    formatter.timeStyle = .MediumStyle  
    timeLabel.text = formatter.stringFromDate(clock.currentTime)  
}
```

- Replace the observer removal in `deinit` with an invalidation of the timer.

```
deinit {  
    if let timer = self.timer {  
        timer.invalidate()  
    }  
}
```

- Explain the best practice of invalidating a scheduled timer in a deinitializer, and how the optional binding of the `timer` property ensures that its value is not `nil` before calling `invalidate`.
- Run the app (⌘R) and observe that it continuously displays the current time.

Closing

What happens when we rotate the app (⌘→) in the Simulator?

Modifications And Extensions

- Consider how the updating of time is a model concern, and refactor the updating of the model's time as a property that is continuously updated with an `NSTimer` internal to the model. Use key-value observing to then update the label when the model's time changes.

Resources

iOS App Programming Guide: The Main Run Loop https://developer.apple.com/library/ios/documentation/iPhone/Conceptual/iPhoneOSProgrammingGuide/TheAppLifeCycle/TheAppLifeCycle.html#//apple_ref/doc/uid/TP40007072-CH2-SW14

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

`NSTimer` Class Reference https://developer.apple.com/library/ios/documentation/Cocoa/Reference/Foundation/Classes/NSTimer_Class/index.html

Threading Programming Guide: Run Loops <https://developer.apple.com/library/ios/documentation/Cocoa/Conceptual/Multithreading/RunLoopManagement/RunLoopManagement.html>

Timer Programming Topics <https://developer.apple.com/library/ios/documentation/Cocoa/Conceptual/Timers/>

The Swift Programming Language: Optional Binding https://developer.apple.com/library/prerelease/ios/documentation/Swift/Conceptual/Swift_Programming_Language/TheBasics.html#//apple_ref/doc/uid/TP40014097-CH5-ID333