

UI Design and Storyboarding

Mobile Application Development in iOS

School of EECS

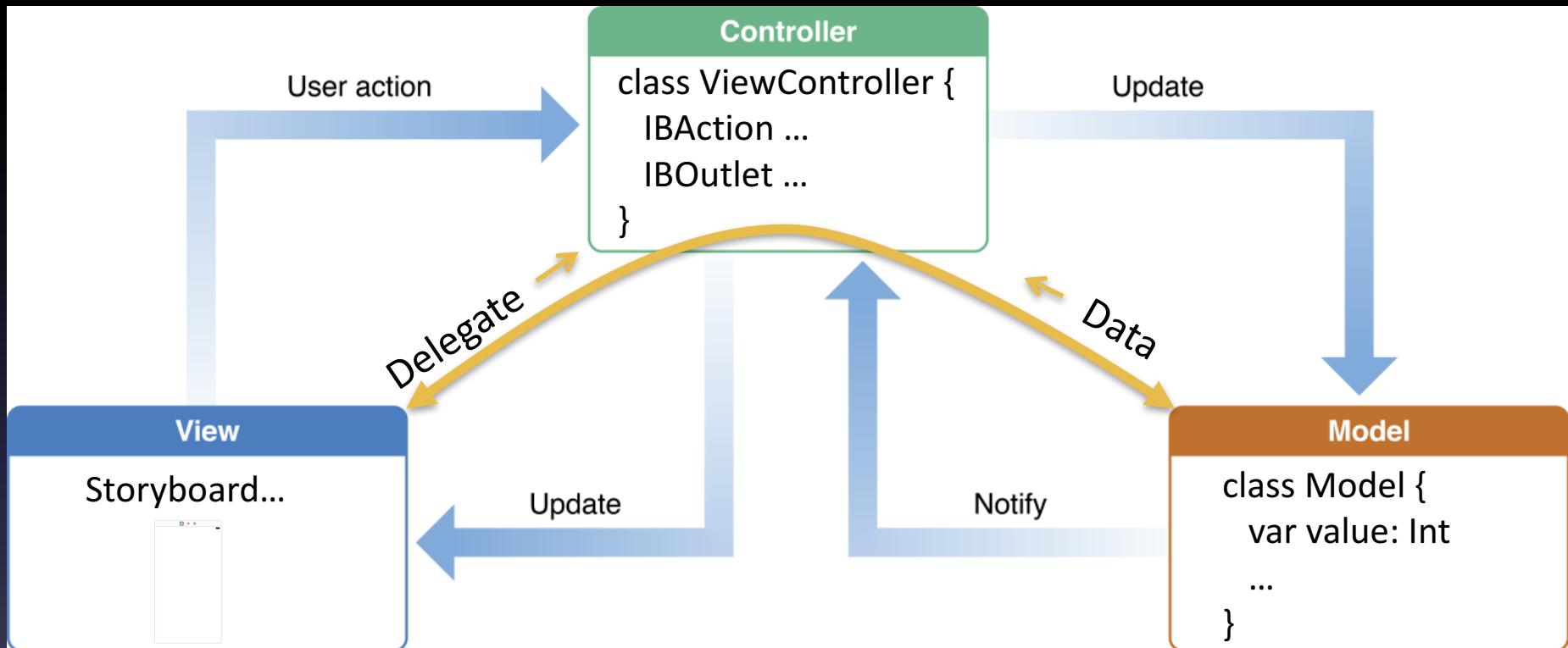
Washington State University

Instructor: Larry Holder

Outline

- Model-View-Controller (MVC) design
- Storyboarding
- Delegates
- Multi-threading and timers

Model-View-Controller (MVC)



Model

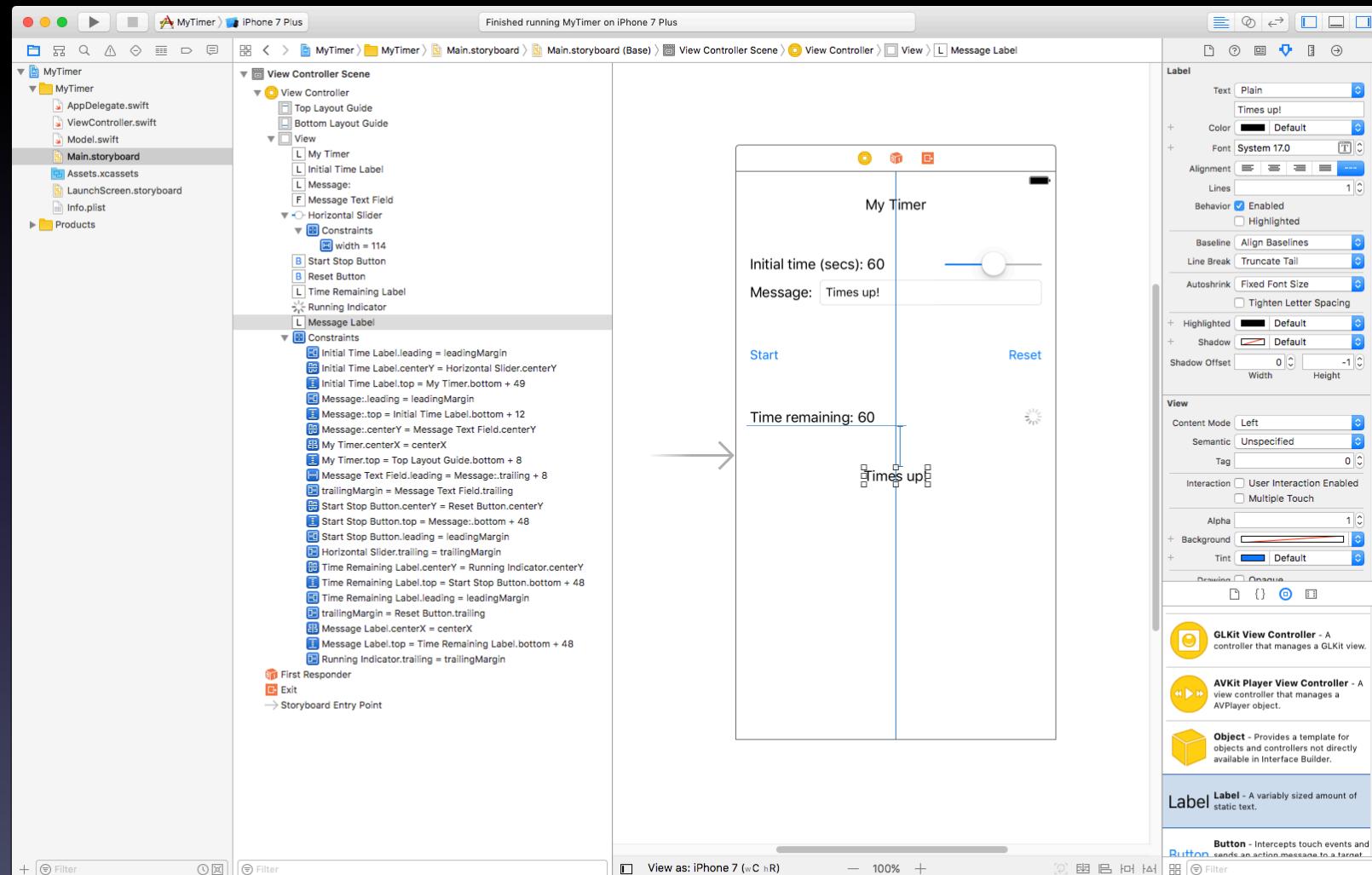
```
class MyTimer {  
    var initial: Int  
    var current: Int  
    var message: String  
    var running: Bool  
  
    init (initial: Int, message: String) {  
        self.initial = initial  
        self.current = initial  
        self.message = message  
        self.running = false  
    }  
  
    func start () {  
        self.running = true  
    }  
  
    func stop () {  
        self.running = false  
    }  
}
```

Model (cont.)

```
...
func reset () {
    self.running = false
    self.current = self.initial
}

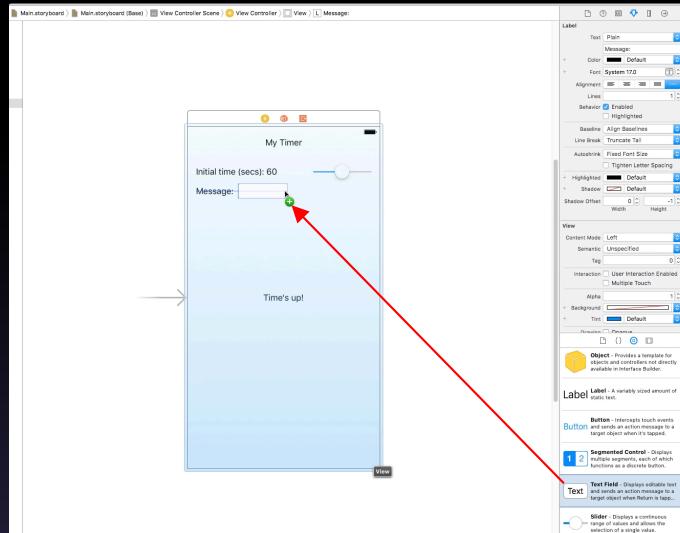
// Decrements timer if running; returns true if time's up
func decrement () -> Bool {
    if (self.running) {
        current = current - 1
        if (current == 0) {
            self.running = false
            return true
        }
    }
    return false
}
```

Storyboard

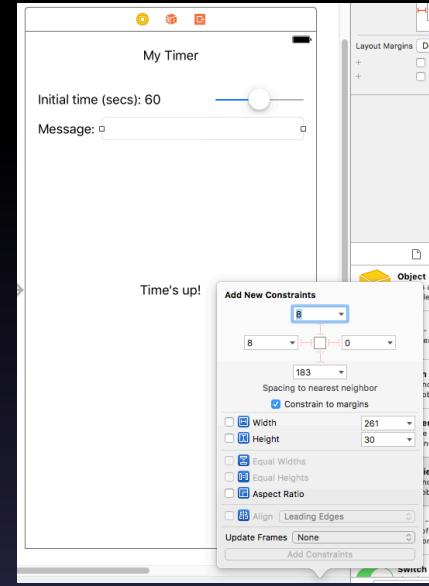


Storyboard: Adding Elements

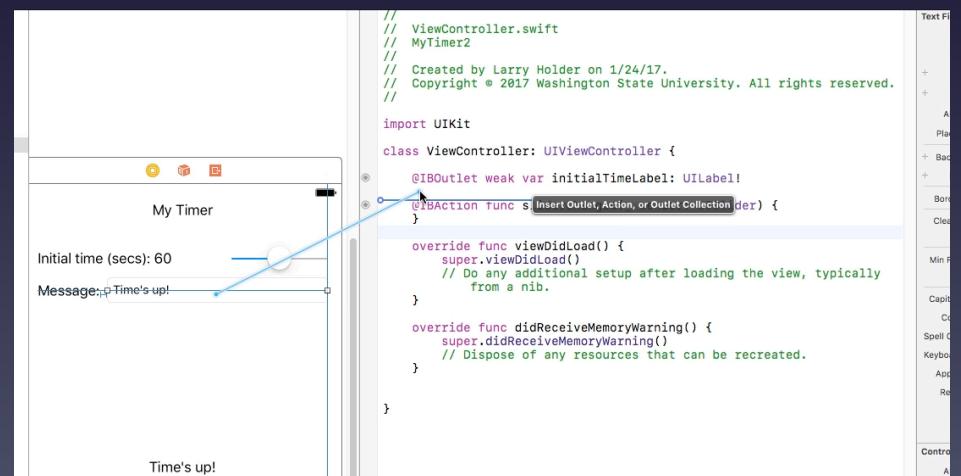
1. Drag element into view.



2. Add layout constraints.



3. Connect element to view controller.



- Adding elements to views
 - Create **IBOutlet** to get/set properties of element
 - Create **IBAction** to detect interaction with element

Delegate

- Object that responds to events from another object
- Defines a protocol that must be followed
 - Required methods
 - Optional methods
- View elements generating multiple different actions use delegates (rather than IBAction)

Delegate Example: UITextField

- UITextFieldDelegate
 - textFieldDidBeginEditing
 - textFieldDidEndEditing
 - textFieldShouldReturn
- More
 - developer.apple.com/reference/uikit/uitextfielddelegate

UITextField Delegate

```
class ViewController: UIViewController, UITextFieldDelegate {
    @IBOutlet weak var messageTextField: UITextField!
    @IBOutlet weak var timesUpMessage: UILabel!

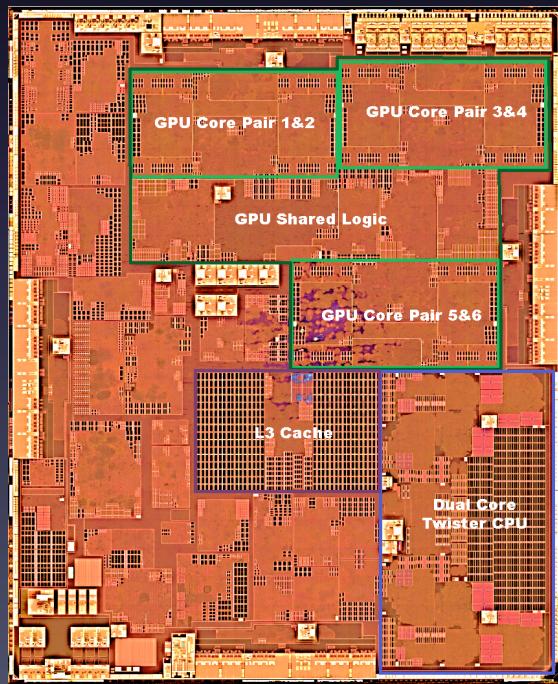
    func textFieldDidEndEditing(_ textField: UITextField) {
        timesUpMessage.text = textField.text
    }

    func textFieldShouldReturn(_ textField: UITextField) -> Bool {
        textField.resignFirstResponder() // remove keyboard on Return
        return false // do default behavior? (no difference here)
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        messageTextField.delegate = self
    }
}
```

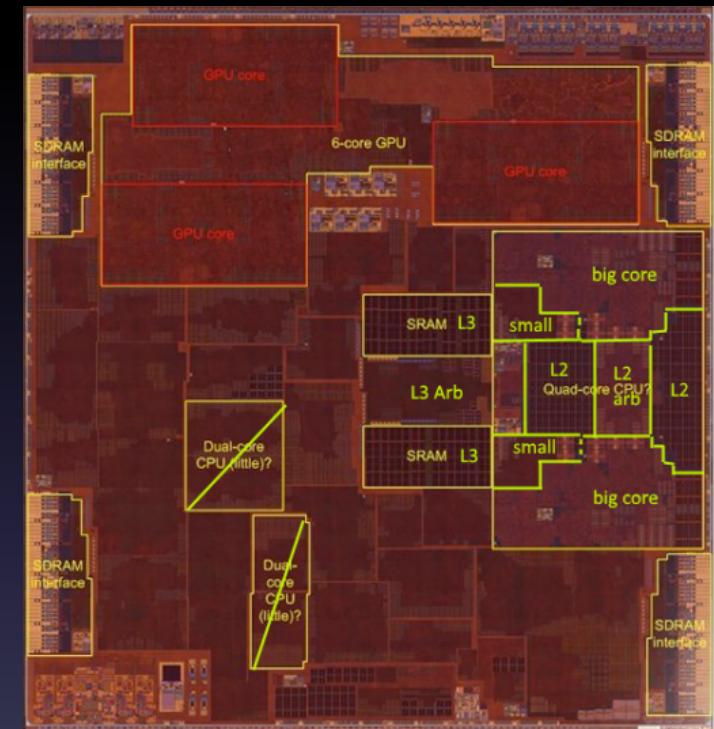
Multi-Threading

- Running multiple tasks concurrently
- Tasks assigned to threads
- Threads assigned to processor cores



A9 Processor

- 2 cores
- iPhone 6



A10 Processor

- 4 cores
- iPhone 7

Multi-Threading in iOS

- Referred to as “Concurrency”
 - Manipulate “queues”, not threads
- Grand Central Dispatch (GCD)
 - iOS mechanism for sending tasks to different queues
 - You can create your own queues, and assign tasks to them
 - Threads take tasks from queues and run on a core
- Main queue used for UI and user interaction
 - Don’t put intensive tasks on Main queue!
- Also, UI changes not on Main queue “lost”

```
let myQueue = DispatchQueue.global()
myQueue.async {
    // Do something, not affecting UI
}
```

```
DispatchQueue.main.async {
    // Do something easy
    // Or affecting UI
}
```

Timers

- Timer class
 - Wait specified time interval, then call function
 - Can repeat
- Start immediately

```
scheduledTimer(withDuration: TimeInterval,  
           repeats: Bool, block: @escaping (Timer) -> Void) -> Timer
```

- Stop `Timer.invalidate()`

Timers

- Create

```
init(timeInterval interval: TimeInterval,  
      repeats: Bool, block: @escaping (Timer) -> Void) -> Timer
```

- Create to fire later

```
init(fire: Date, timeInterval interval: TimeInterval,  
      repeats: Bool, block: @escaping (Timer) -> Void) -> Timer
```

- Add to run loop

```
let runLoop = RunLoop.current  
runLoop.add(timer, forMode: .defaultRunLoopMode)
```

Timers

```
var timer1: Timer!
var timer2 = Timer.init(timeInterval: 1.0, repeats: false, block:
handleTick)

func startButtonTapped {
    timer1 = Timer.scheduledTimer(withTimeInterval: 1.0,
        repeats: true, block: handleTick)
    RunLoop.current.add(timer2, forMode: .defaultRunLoopMode)
}

func stopButtonTapped {
    timer1.invalidate()
    timer2.invalidate()
}

func handleTick (timer: Timer) {
    print("tick")
}
```

Resources

- Storyboard
 - <http://help.apple.com/xcode/mac/8.3/>
- Protocols and Delegates
 - https://developer.apple.com/library/content/documentation/Swift/Conceptual/Swift_Programming_Language/Protocols.html
- UITextField and UITextFieldDelegate
 - <https://developer.apple.com/reference/uikit/uitextfield>
 - <https://developer.apple.com/reference/uikit/uitextfielddelegate>
- Grand Central Dispatch (GCD)
 - <https://developer.apple.com/reference/dispatch>
- Timer
 - <https://developer.apple.com/reference/foundation/timer>