

Gestures

Mobile Application Development in iOS









School of EECS

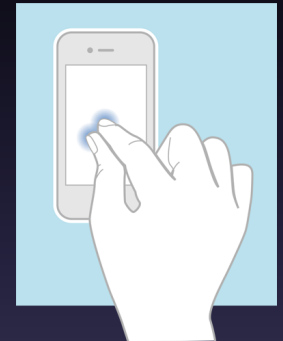
Washington State University

Instructor: Larry Holder

Outline

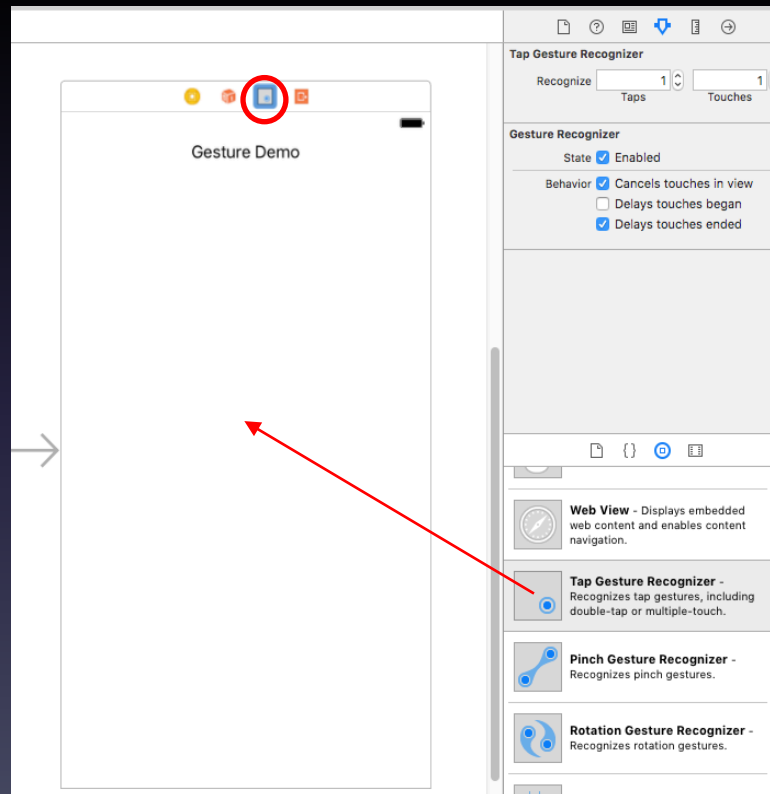
- Gestures
- Gesture recognizers
- Responder chain
- Gesture states

	Tap Gesture Recognizer - Recognizes tap gestures, including double-tap or multiple-touch.
	Pinch Gesture Recognizer - Recognizes pinch gestures.
	Rotation Gesture Recognizer - Recognizes rotation gestures.
	Swipe Gesture Recognizer - Recognizes swipe gestures.
	Pan Gesture Recognizer - Recognizes pan (dragging) gestures.
	Screen Edge Pan Gesture Recognizer - Recognizes pan (dragging) gestures that start near the screen edge.
	Long Press Gesture Recognizer - Recognizes long press gestures, based on the number of times the gesture is performed.
	Custom Gesture Recognizer - Recognizes custom gestures. Set a custom subclass in the Identity Inspector.



Add Gesture in Storyboard

- Step 1: Drag gesture into view



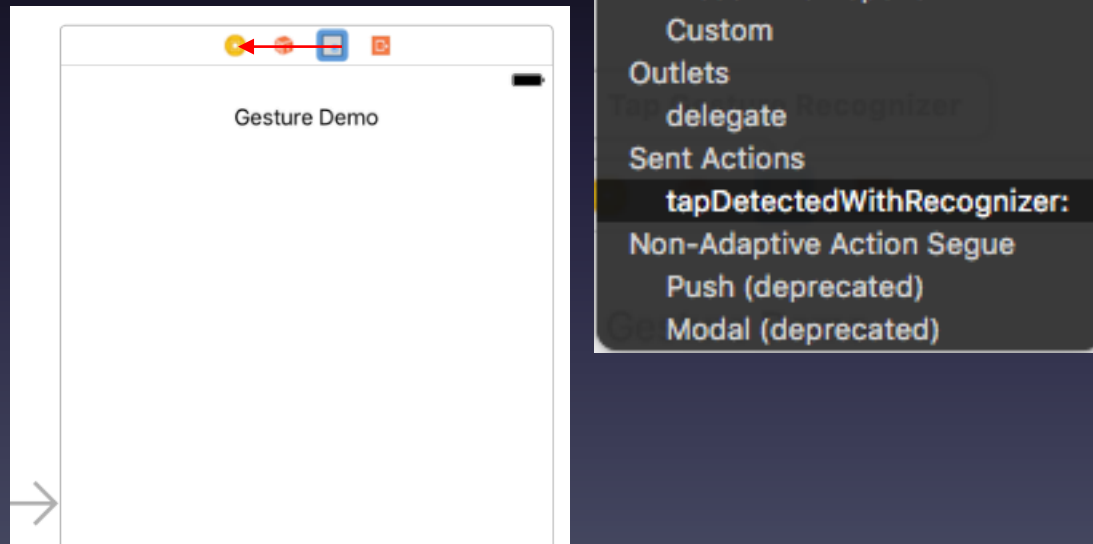
Add Gesture in Storyboard

- Step 2: Implement **@IBAction** for gesture

```
class ViewController: UIViewController {  
  
    @IBAction func tapDetected (recognizer: UITapGestureRecognizer)  
    {  
        let point = recognizer.location(in: self.view)  
        let xi = Int(point.x)  
        let yi = Int(point.y)  
        print("tap detected at \(xi), \(yi)")  
    }  
  
}
```

Add Gesture in Storyboard

- Step 3: Connect gesture to action



Add Gesture Programmatically

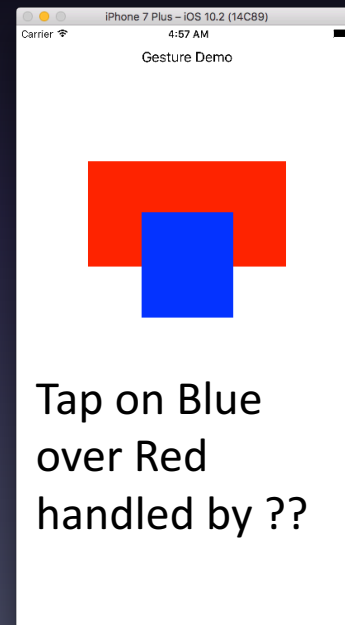
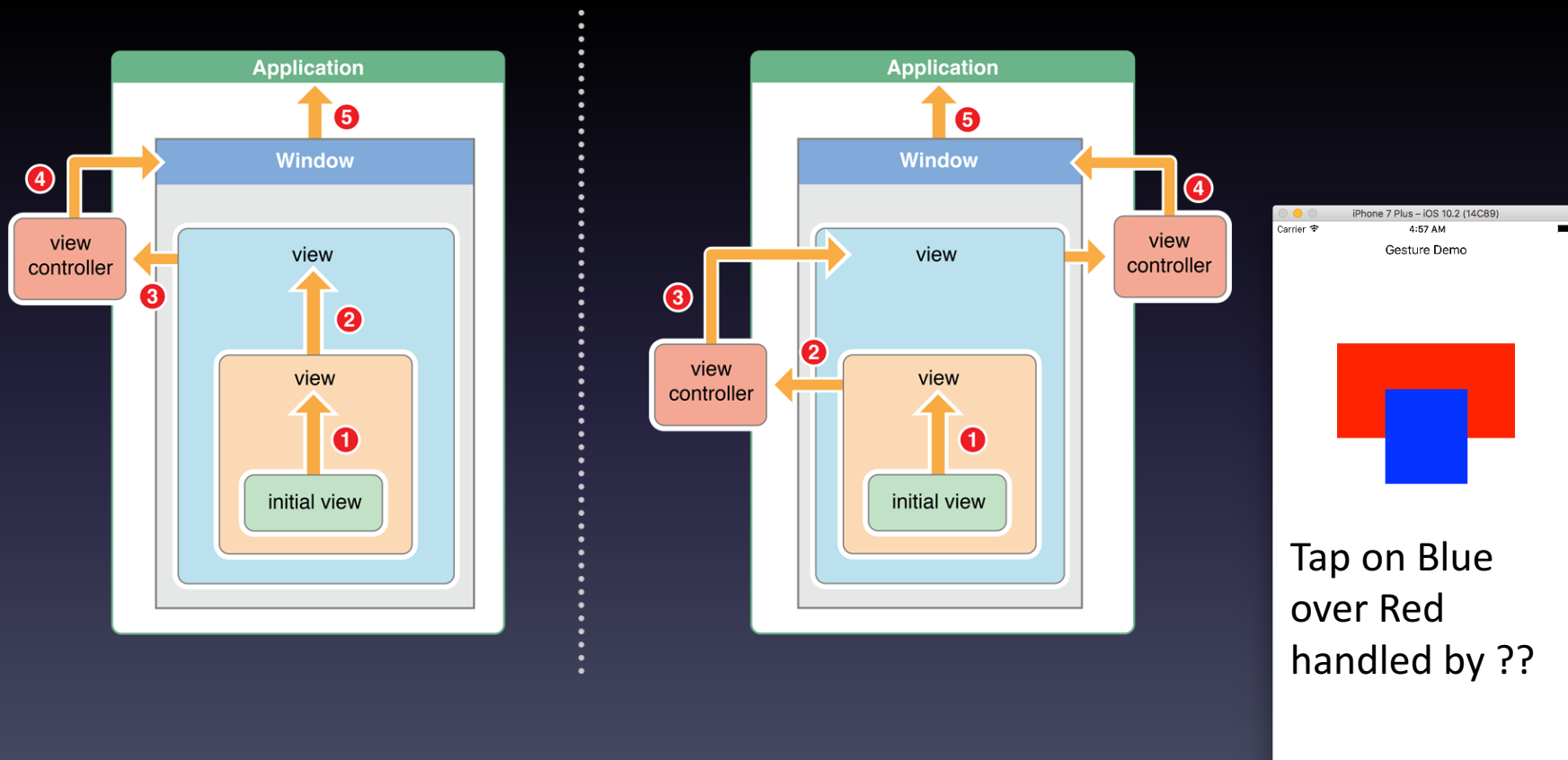
- View controller class conforms to `UIGestureRecognizerDelegate`
- Implement method to handle gesture
- Create gesture recognizer and add to view

Add Gesture Programmatically

```
class ViewController: UIViewController, UIGestureRecognizerDelegate
{
    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view
        let tapGestureRecognizer = UITapGestureRecognizer(target: self,
                                                         action: #selector(handleTap))
        tapGestureRecognizer.delegate = self
        self.view.addGestureRecognizer(tapGestureRecognizer)
    }









    func handleTap (recognizer: UITapGestureRecognizer) {
        let point = recognizer.location(in: self.view)
        let xi = Int(point.x)
        let yi = Int(point.y)
        print("tap detected at \(xi), \(yi)")
    }
}
```

Responder Chain: Route of a Touch Event



Other Gestures

- UITapGestureRecognizer (multiple taps/touches)
- UIPinchGestureRecognizer
- UIRotationGestureRecognizer
- UISwipeGestureRecognizer
- UIPanGestureRecognizer
- UIScreenEdgeGestureRecognizer
- UILongPressGestureRecognizer
- UIGestureRecognizer (custom)

	Tap Gesture Recognizer - Recognizes tap gestures, including double-tap or multiple-touch.
	Pinch Gesture Recognizer - Recognizes pinch gestures.
	Rotation Gesture Recognizer - Recognizes rotation gestures.
	Swipe Gesture Recognizer - Recognizes swipe gestures.
	Pan Gesture Recognizer - Recognizes pan (dragging) gestures.
	Screen Edge Pan Gesture Recognizer - Recognizes pan (dragging) gestures that start near the screen edge.
	Long Press Gesture Recognizer - Recognizes long press gestures, based on the number of times the gesture is performed.
	Custom Gesture Recognizer - Recognizes custom gestures. Set a custom subclass in the Identity Inspector.

Gesture States (e.g., Pan)

```
let panGestureRecognizer = UIPanGestureRecognizer(target: self,
                                                    action: #selector(handlePan))
panGestureRecognizer.delegate = self
self.view.addGestureRecognizer(panGestureRecognizer)

func handlePan (recognizer: UIPanGestureRecognizer) {
    let point = recognizer.location(in: self.view)
    let xi = Int(point.x)
    let yi = Int(point.y)
    if (recognizer.state == .began) {
        print("pan began at \(xi), \(yi)")
    }
    if (recognizer.state == .changed) {
        print("pan moved to \(xi), \(yi)")
    }
    if (recognizer.state == .ended) {
        print("pan ended at \(xi), \(yi)")
    }
}
```

Multi-Touch Gestures

- Set `UITapGestureRecognizer.numberOfTouchesRequired`

```
let twoTouchGestureRecognizer = UITapGestureRecognizer(target: self,
                                                         action: #selector(handleTwoTouch))
twoTouchGestureRecognizer.delegate = self
twoTouchGestureRecognizer.numberOfTouchesRequired = 2
self.view.addGestureRecognizer(twoTouchGestureRecognizer)

func handleTwoTouch (recognizer: UITapGestureRecognizer) {
    let point = recognizer.location(in: self.view)
    let x = Int(point.x)
    let y = Int(point.y)
    print("two touch detected at (\(x), \(y))")
}
```

Multi-Touch Gestures

- Access more than one touch with `touchesBegan`
- Don't forget to set view's `isMultipleTouchEnabled`

```
self.view.isMultipleTouchEnabled = true

override func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?) {
    let touchCount = touches.count
    print("detected \(touchCount) touches")
    for touch in touches {
        let point = touch.location(in: self.view)
        let x = Int(point.x)
        let y = Int(point.y)
        print("    location (\(x), \(y))")
    }
}
```

Custom Gestures

- Create subclass of `UIGestureRecognizer`
- Import `UIKit.UIGestureRecognizerSubclass`
 - So `state` can be changed
- Override main gesture functions
 - `touchesBegan(_ touches: Set<UITouch>, with event: UIEvent)`
 - `touchesMoved(_ touches: Set<UITouch>, with event: UIEvent)`
 - `touchesEnded(_ touches: Set<UITouch>, with event: UIEvent)`
 - `touchesCanceled(_ touches: Set<UITouch>, with event: UIEvent)`
 - `reset()`

Resources

- **UIGestureRecognizer API Reference**
 - <https://developer.apple.com/reference/uikit/uigestureRecognizer>
- **Event Handling Guide (still in Obj-C)**
 - <https://developer.apple.com/library/prerelease/content/documentation/EventHandling/Conceptual/EventHandlingiPhoneOS/Introduction/Introduction.html>