Gestures

Mobile Application Development in iOS

School of EECS

Washington State University

Instructor: Larry Holder
Outline

- Gestures
- Gesture recognizers
- Gesture states
- Custom gestures
Add Gesture in Storyboard

• Step 1: Drag gesture into view
Add Gesture in Storyboard

• Step 2: Connect gesture to @IBAction

```swift
// In ViewController class...

@IBAction func tapDetected (_ sender: UIGestureRecognizer) {
    let point = sender.location(in: self.view)
    let x = Int(point.x)
    let y = Int(point.y)
    print("tap detected at (\(x), \(y))")
}
```
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 twoTouchTapGestureRecognizer = UITapGestureRecognizer(target: self,
            action: #selector(handleTwoTouchTap))
        twoTouchTapGestureRecognizer.delegate = self
        twoTouchTapGestureRecognizer.numberOfTouchesRequired = 2
        self.view.addGestureRecognizer(twoTouchTapGestureRecognizer)
    }

    @objc func handleTwoTouchTap (_ sender: UITapGestureRecognizer) {
        let point = sender.location(in: self.view)
        let x = Int(point.x)
        let y = Int(point.y)
        print("two-touch tap detected at (\(x), \(y))")
    }
}

Mobile Application Development in iOS 6
Other Gestures: Subclasses of UIGestureRecognizer

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

Note: Only one gesture detected per user interaction.
Gesture States

- `(UIGestureRecognizer.State)
  - `.possible` (default)
  - `.began` (calls action)
  - `.changed` (calls action)
  - `.ended` (calls action, reset to `.possible`)
  - `.cancelled` (calls action, reset to `.possible`)
  - `.failed` (no call to action, reset to `.possible`)
  - `.recognized` (multi-touch, calls action, reset to `.possible`)

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

@objc func handlePan (_ sender: UIPanGestureRecognizer) {
    let point = sender.location(in: self.view)
    let x = Int(point.x)
    let y = Int(point.y)
    if (sender.state == .began) {
        print("pan began at (\(x), \(y))")
    }
    if (sender.state == .changed) {
        print("pan moved to (\(x), \(y))")
    }
    if (sender.state == .ended) {
        print("pan ended at (\(x), \(y))")
    }
}
Custom Gestures

• Create subclass of `UIGestureRecognizer`

• Import `UIKit.UIGestureRecognizerSubclass`
  – Defines methods and properties to override

• 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)`
  – `touchesCancelled(_ touches: Set<UITouch>, with event: UIEvent)`
  – `reset()`
Custom Gesture Example: Backslash
import UIKit
import UIKit.UIGestureRecognizerSubclass

class BackslashGestureRecognizer: UIGestureRecognizer {

    var initialPoint: CGPoint!
    var previousPoint: CGPoint!

    override func touchesBegan(_ touches: Set<UITouch>,
                                 with event: UIEvent) {
        print("backslash: touchesBegan")
        let touch = touches.first
        if let point = touch?.location(in: self.view) {
            initialPoint = point
            previousPoint = point
            state = .began
        }
    }
}
override func touchesMoved(_ touches: Set<UITouch>,
    with event: UIEvent) {
    print("backslash: touchesMoved")
    let touch = touches.first
    if let point = touch?.location(in: self.view) {
        if ((point.x == previousPoint.x) &&
            (point.y == previousPoint.y)) {
            previousPoint = point
            state = .changed
        } else {
            state = .failed
        }
    }
}
override func touchesEnded(_ touches: Set<UITouch>,
    with event: UIEvent) {
    print("backslash: touchesEnded")
    let touch = touches.first
    if let point = touch?.location(in: self.view) {
        if (point != initialPoint) {
            state = .ended
        } else {
            state = .failed
        }
    }
}
override func touchesCancelled(_ touches: Set<UITouch>,
    with event: UIEvent) {
    print("backslash: touchesCancelled")
    state = .cancelled
}

override func reset() {
    print("backslash: reset")
}
}
Backslash Custom Gesture (5)

```swift
// In viewDidLoad...
let backslashGestureRecognizer = 
    BackslashGestureRecognizer(target: self, 
                                 action: #selector(handleBackslash))
backslashGestureRecognizer.delegate = self
self.view.addGestureRecognizer(backslashGestureRecognizer)

// In ViewController...
@objc func handleBackslash(_ sender: BackslashGestureRecognizer) {
    if sender.state == .ended {
        print("backslash detected")
    }
}
```

Remember to conform ViewController to UIGestureRecognizerDelegate
var boxViews: [UIView] = []

func drawBox(_ point: CGPoint) {
    let boxRect = CGRect(x: point.x, y: point.y,
        width: 5.0, height: 5.0)
    let boxView = UIView(frame: boxRect)
    boxView.backgroundColor = UIColor.red
    self.view?.addSubview(boxView)
    boxViews.append(boxView)
}

func clearBoxes() {
    for boxView in boxViews {
        boxView.removeFromSuperview()
    }
    boxViews.removeAll()
}
Multiple Gestures

• Simultaneous gestures
  – func gestureRecognizer(_ gestureRecognizer: UIGestureRecognizer, 
    shouldRecognizeSimultaneouslyWith otherGestureRecognizer: 
    UIGestureRecognizer) -> Bool

• Gesture preference
  – func gestureRecognizer(_ gestureRecognizer: UIGestureRecognizer, 
    shouldRequireFailureOf otherGestureRecognizer: 
    UIGestureRecognizer) -> Bool
func gestureRecognizer(_ gestureRecognizer: UIGestureRecognizer, shouldRequireFailureOf otherGestureRecognizer: UIGestureRecognizer) -> Bool {
    if gestureRecognizer is BackslashGestureRecognizer {
        if otherGestureRecognizer is UITapGestureRecognizer {
            return true
        }
    }
    return false
}

func gestureRecognizer(_ gestureRecognizer: UIGestureRecognizer, shouldRecognizeSimultaneouslyWith otherGestureRecognizer: UIGestureRecognizer) -> Bool {
    if gestureRecognizer is BackslashGestureRecognizer {
        if otherGestureRecognizer is UITapGestureRecognizer {
            return true
        }
    }
    return false
}
Multiple Gestures

• Preserving interactions with view elements
  – E.g., button taps will go to gesture, not UIButton
  – Use `gestureRecognizer: shouldReceive`
Resources

• Human Interface Guidelines: Gestures
  – developer.apple.com/design/human-interface-guidelines/ios/user-interaction/gestures

• UIGestureRecognizer API Reference
  – developer.apple.com/documentation/uikit/uigesturerecognizer

• Implementing a custom gesture recognizer
  – developer.apple.com/documentation/uikit/touches_presses_and_gestures/implementing_a_custom_gesture_recognizer