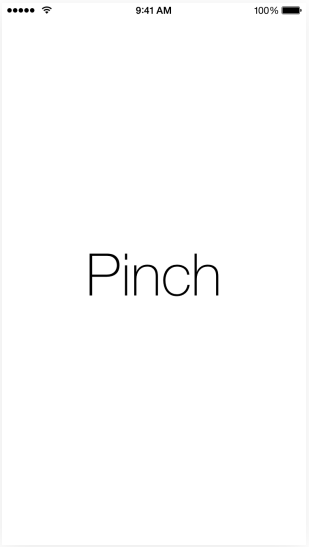
Gesturizer
Lesson 6

Description

Add a Pinch Gesture Recognizer and check the gesture recognizer state to determine when to update the label.



Learning Outcomes

- Apply a Pinch Gesture Recognizer to detect a pinch gesture with multiple touches.
- Discover the `state` property of gesture recognizers, and revise an event handler to inspect this state to properly update an interface.

Vocabulary

pinch gesture	Pinch Gesture Recognizer	`UIGestureRecognizer`
enumeration	`UIGestureRecognizerState`	

Materials

- **Gesturizer Lesson 6** Xcode project
- **Enumerations** presentation

Opening

How are some gestures that involve multiple touches, and how do you think we can get our app to recognize them?

Agenda

- Using Interface Builder and the Object Library (⌥⌘L), drag a Pinch Gesture Recognizer into the Document Outline (◧).
- Using the Assistant Editor (⌥⌘↩), Control-drag a connection from the Pinch Gesture Recognizer to a new controller action called `pinch:`.

```
@IBAction func pinch(sender: UIPinchGestureRecognizer) {
    showGestureName("Pinch")
}
```

- Using the Interface Builder Document Outline (◧), Control drag a from the View to the Pinch Gesture Recognizer to add the Pinch Gesture Recognizer to the View's `gestureRecognizers` outlet collection.
- Run the app (⌘R), hold down the ⌥ key to simulate two fingers, click and drag the mouse to simulate a pinch, and observe how the **Pinch** label flickers.
- Discuss when, and how frequently, the Pinch Gesture Recognizer must be calling the controller `pinch:` method.
- Discuss the requirement of showing the **Pinch** label only when the pinch gesture has completed.
- Using the Xcode Documentation and API Reference (⇧⌘0), examine the `UIGestureRecognizer` class reference, its `state` property, and the `UIGestureRecognizerState` enumeration.
- Discuss the documented `UIGestureRecognizerState` values, such as `Possible` and `Began`.
- Present the concept of enumerations.
- In the `ViewController` class, update the implementation of the `pinch:` method.

```
@IBAction func pinch(sender: UIPinchGestureRecognizer) {
    if sender.state == .Ended {
        showGestureName("Pinch")
    }
}
```

- Discuss how Swift infers the type of the `state` property as a `UIGestureRecognizerState`, which allows for the shorthand syntax of `.Ended`.
- Run the app (⌘R), hold down the ⌥ key to simulate two fingers, click and drag the mouse to simulate a pinch, and observe the **Pinch** text appear when the mouse button is released.

Closing

How might you make the words **Pinch Began** and **Pinch Ended** appear when the pinch gesture begins and ends?

Modifications And Extensions

- Investigate the other `UIGestureRecognizer` properties, and use the `delaysTouchesEnded` property to delay the display of the **Pinch** label.

Resources

Event Handling Guide for iOS https://developer.apple.com/library/ios/documentation/EventHandling/Conceptual/EventHandlingiPhoneOS/Introduction/Introduction.html

Creating an Action Connection https://developer.apple.com/library/ios/recipes/xcode_help-IB_connections/chapters/CreatingAction.html

Cocoa Core Competencies: Target-Action http://developer.apple.com/library/ios/documentation/General/Conceptual/Devpedia-CocoaApp/TargetAction.html

UIGestureRecognizer Class Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIGestureRecognizer_Class/index.html

UIPinchGestureRecognizer Class Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIPinchGestureRecognizer_Class/index.html