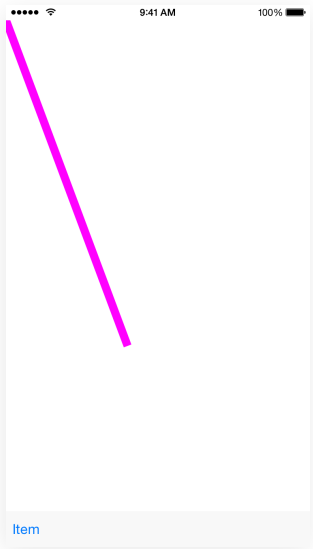
FingerPainter
Lesson 3

Description

Capture point coordinates within the touch event handlers, and inspect the x and y components of the touch coordinates using custom breakpoints.



Learning Outcomes

- Recognize how touch events can send multiple `UITouch` objects to event handlers.
- Discover how to obtain point coordinates from `UITouch` objects.
- Observe the use of the Swift forced type cast operator.
- Practice creating custom breakpoint actions to print console messages.

Vocabulary

event	set (data structure)	`Set<NSObject>`
`UITouch`	`CGPoint`	type cast operator
breakpoint		

Materials

- **FingerPainter Lesson 3** Xcode project

Opening

How can we use the touch events to obtain coordinates for drawing?

Agenda

- Explain that, because iOS can respond to multiple touches (with multiple fingers), a `Set` of `UITouch` objects is passed to `touchesBegan:withEvent:` and `touchesMoved:withEvent:`.
- Update the implementation of `touchesBegan:withEvent:`.

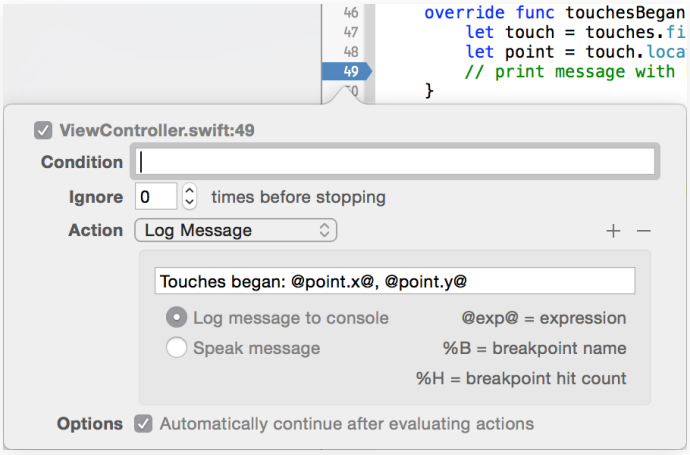
```
override func touchesBegan(touches: Set<NSObject>,
   withEvent event: UIEvent) {
   let touch = touches.first as! UITouch
   let point = touch.locationInView(view)
   // print message with breakpoint here
}
```

- Using the Xcode Documentation and API Reference (⇧⌘0) explore the `UITouch` class reference, drawing attention to the `locationInView:` method.
- Explain that `touchesBegan:withEvent:` is passed a `Set` of objects, `touches`, and how we call `first` upon touches to retrieve the first object in the `Set`.
- Explain that, because the `touches` set has a placeholder type of `NSObject`, the code uses the Swift forced type cast `as!` operator to convert the `first` object in the `touches` set to a `UITouch` object.
- Discuss how all `ViewController` objects have an inherited `view` property, and how the `touchesBegan:withEvent:` method obtains a `CGPoint` , representing a coordinate within the `view`, from the `UITouch` object.
- Using the Xcode Documentation and API Reference (⇧⌘0), explore the `CGPoint` structure.
- Update the implementation of `touchesMoved:withEvent:`.

```
override func touchesMoved(touches: Set<NSObject>,
   withEvent event: UIEvent) {
   let touch = touches.first as! UITouch
   let point = touch.locationInView(view)
   // print message with breakpoint here
}
```

- Discuss how `touchesMoved:withEvent:` will be called multiple times while the finger drags across the screen, and how the `CGPoint` obtained from the touch events can be used to draw a line from point to point as the finger moves.

- Add custom breakpoints to the bodies of both the `touchesBegan:withEvent:` and `touchesMoved:withEvent:` methods that use a **Log Message** action to print the `x` and `y` components of the `CGPoint`, and automatically continue.



- Run the app (⌘R), click the screen to simulate a touch, and observe the coordinates printed on the console (⇧⌘C). Click and drag on the screen to simulate a finger moving across the screen, and observe the coordinates printed on the console(⇧⌘C) by `touchesMoved:withEvent:`.

Closing

Since we are only simulating a single touch, the `touches` set only contains one `UITouch` object. What do you think happens to the set when we touch the screen with multiple fingers?

Modifications And Extensions

- Investigate the `UIEvent` class reference, and add the time of the touches to the breakpoint log messages.
- Investigate how to detect multiple touches (multiple fingers) on the screen, and print the locations of each touch on the console.

Resources

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

Setting Breakpoint Actions and Options http://developer.apple.com/library/ios/recipes/xcode_help-breakpoint_navigator/articles/setting_breakpoint_actions_and_options.html

UIResponder Class Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIResponder_Class/index.html

The Swift Programming Language: Collection Types https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/CollectionTypes.html

UITouch Class Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UITouch_Class/index.html

CGGeometry Reference https://developer.apple.com/library/prerelease/ios/documentation/GraphicsImaging/Reference/CGGeometry/index.html