GENERAL ASSEMBLY

MOBILE DEVELOPMENT
INTRO TO FUNCTIONS

Rudd Taylor
Founder, SALT

LEARNING OBJECTIVES

- Identify functions and implement best practices
- Create hooks from interface builder to Swift code

REVIEW LESSON 3

GETTING STARTED

INTRO TO FUNCTIONS

WHAT IS A FUNCTION?

- A function is a series of repeatable steps that, at some point, ends
- Optional input and output
- Multiple inputs and outputs, as needed

CALLING FUNCTIONS

- ***name***() // No parameters, no return
- ***name***(***parameter***) // One parameter, no return
- ***name***(***parameter, parameterTwoName: parameterTwo***) // Two parameters, no return
- var result = ***name***(***parameter***) // One parameter, one returned value
- let result = ***name***() {/* code */} // No parameters, two returned values
  - println("\(result.paramOneName) \(result.paramTwoName)")

DEFINING FUNCTIONS

- func ***name***() { /* code */} // No parameters, no return
- func ***name***(***parameterName: type***) { /* code */} // One parameter, no return
- func ***name***(***parameterName: type, parameterTwoName: type***) { /* code */} // Two parameters, no return
- func ***name***(***parameterName: type***) -> ***returnType*** { /* code */} // One parameter, one returned value
- func ***name***() -> (***returnOne: valueOne***, ***returnTwo: valueTwo***) {/* code */} // No parameters, two returned values

XCODE DEMO: FUNCTIONS

FUNCTIONS RECAP

- Be descriptive: Name your functions with descriptive names and descriptive parameters
- Be brief: Keep your functions short (i.e. approximately less than a screen's worth of content). You should be able to describe what they do in once sentence
- Compose: Your functions can call each other
- DRY: Don't repeat yourself. Any time you find the urge to copy and paste, there may be an opportunity to break into a function

WHEN TO USE FUNCTIONS

- Functions are VERY common building blocks when writing code
  - But figuring out how to break them up is HARD, even for intermediate developers
- Any time you find the urge to copy and paste
- Any time you have multiple parts of your application sharing the same functionality, or very similar functionality with different parameters
- KISS: Avoid the urge to over-compose. Over-composed code can be just as difficult to read as under-composed code

HOOKING UP INTERFACE BUILDER TO CODE

STORYBOARDS

- Remember storyboards?
- Our view controllers in storyboards can be (and usually are) represented in code
- Our code can modify those view controllers, change its views, the properties of those views, etc
- We create the connections between our view controllers using ‘outlets’

XCODE DEMO: STORYBOARDS