Announcements

- **Status update 2 presentations start on Monday**

Today's Topics

- **Swift**
  - Overview
  - Syntax
  - Examples
- **Xcode 6**
  - Playgrounds
- **Swift code in Objective-C**
- **Objective-C code in Swift**

Swift

- New programming language developed by Apple
- Announced at WWDC 2014
- Interoperates with Objective-C
  - Both are considered first class citizens
- Still a work in progress

Hello World in Swift

```
println("Hello World")
```

- No semicolons
- No main method needed

Variables and Constants

- **Swift uses var and let to describe variables and constants**
- **Variables and constants have a type**
  - let languageName: String = "Swift"
  - var version: Double = 1.0
  - let isEverChanging: Bool = true
- **Swift supports type inference**
  - let languageName = "Swift" //inferred as String
  - var version = 1.0 //inferred as Double
  - let isEverChanging = true //inferred as Bool

Strings

- **Swift makes working with strings easy**

```
let firstName = "John"
let lastName = "Smith"
let fullName = firstName + " " + lastName
```

- **Enumerating through them is familiar**

```
for character in firstName{
    println(character)
}
```

J
o
h
n

String Interpolation

```
let a =2, b = 3

// "2 times 3 is 6"
let mathResult = "\(a) times \(b) is \(a * b)"
```


Collections - Arrays and Dictionaries

```
var names = ["Bob", "Alice", "Mike", "Jen"]
```

– Inferred as a typed collection of Strings

- I could also be more explicit:

```
var names: [String] = ["Bob", "Alice", "Mike", "Jen"]
```

```
var numberOfLegs = ["ant": 6, "snake": 0, "cow" :4]
```

– Inferred as a typed dictionary of Strings and Ints

- Or I could be more explicit:

```
var numberOfLegs: [String: Int] = ["ant": 6, "snake": 0, "cow" :4]
```

Loops

```
while !done {
  keepDoingSomething()
}
for num in 1...5 {     //Prints from 1 up to and including 5
  println("\(num) times 4 is \(num * 4)")
}

for var i= 1; i<= 12; i++ {
    doSomething(i)
 }
```


Conditionals

```
if legCount == 0 {
   println("Does not walk")
} else if legCount == 1 {
  println("Hopping around")
} else {
   println("I can walk")
}
```

```
switch legCount {
  case 0:
   println("Does not walk")

 case 1, 3, 5, 7:
   println("Limps around")

default:
   println("I can walk")
```

Functions

```
func sayHi() {
   println("Hi")
}
   sayHi()

func sayHi(name: String) {
   println("Hi \(name)!")
}
   sayHi("Bob")
```

```
func sayHi(name: String = "CSE 436") {
   println("Hi \(name)!")
 }
   sayHi()          //Prints Hi CSE 436
   sayHi(name: "Bob")    //Prints Hi Bob
```


Functions

```
func sayHi(name: String = "CSE 436")  -> String {
   return "Hi " + name
}
   let name =  sayHi()     //Name contains "Hi CSE 436"
```

```
func refreshWebSite() -> (Int, String ) {
   // refresh
   return (200, "Success")
}
   let (statusCode, message) =refreshWebSite()
```

Closures

- **Self-contained blocks of functionality that can be passed around**
  - Similar to blocks in Objective-C

```
let displayGreeting = {
   println("Hello Class")
}

let displayGreeting: () -> () = {      //Inferred as this
   println("Hello Class")               //looks very similar to a
}                                      function (named closure)

displayGreeting()
```


Optionals

- **Optionals handle the absence of a value**
  - There is a value and it equals x or there isn't a value

```
var numberOfLegs = ["ant": 6, "snake": 0, "cow" :4]
let possibleNumLegs  = numberOfLegs["goat"]   ???
let possibleNumLegs: Int?  = numberOfLegs["goat"]  //Value or nil

If possibleNumLegs {
    let legCount = possibleNumLegs!    //Use ! to unwrap the optional
   println("Goat has \(legCount) legs")
}
```

- **Shorthand for above, if let**

```
If let legCount = possibleNumLegs {
   println("Goat has \(legCount) legs")
}
```

Classes

```
class Person {
    //properties
    //methods
   //initializers
}
```

- No .h files
- No Universal base class //Can use if needed

Classes - Properties

```
class Person {
  var age = 21    //defines the properties

  var description: String {   //defines a computed property
       get {
         return "You are \(age) years old"
       }
   }
}

let somePerson = Person()    //no alloc needed
println(" Hello, you are \(somePerson.age) years old")
```

More Information about Swift Language

- **Swift Programming Guide**
  - Available on iTunes
- **WWDC 2014 Videos**
  - developer.apple.com

Examples in Playground

Integrating Objective-C with Swift

Integrating Swift with Objective-C