LLVM/Swift Introduction on Android

- Fady Farag, BSc Computer Science @ astAte

Agenda

- Introduction to the basic features of Swift
- Brief overview of the latest features of Swift 5 and SwiftUI
- How to port the Swift runtime to the Android OS

Released in March

Swift 5

Xcode 10.2

Developer Preview

Swift 5.1

Xcode 11

Constants

```
let language: String = "Swift"
let introduced: Int = 2014
let isAwesome: Bool = true
```

Type Inference

```
let language: String = "Swift"
let introduced: Int = 2014
let isAwesome: Bool = true
```

Unicode

```
let instruction = "Beware of the 🐶🐮"
let internationalHarmony = "🇬🇧🇺🇸😍"

let π = 3.1415927
let 鼠标 = "🐭"
```

Array and Dictionary

```
let names = ["Lily", "Santiago", "Justyn", "Aadya", true]



let ages = ["Mohsen": 17, "Amy": 40, "Graham": 5]
```

Type Inference

```
let names = ["Lily", "Santiago", "Justyn", "Aadya"]
// an array of String values

let ages = ["Mohsen": 17, "Amy": 40, "Graham": 5]
```

For-In Loop

Characters

```
let dogString = "Dog?!🐶"
for character in dogString.characters {
   print(character)
}
```

```
D
o
g
?!
🐶
```

Modifying a Dictionary

```
var ages = ["Mohsen": 17, "Amy": 40, "Graham": 5]
ages["Justyn"] = 67     // Adds a new value for "Justyn"
```

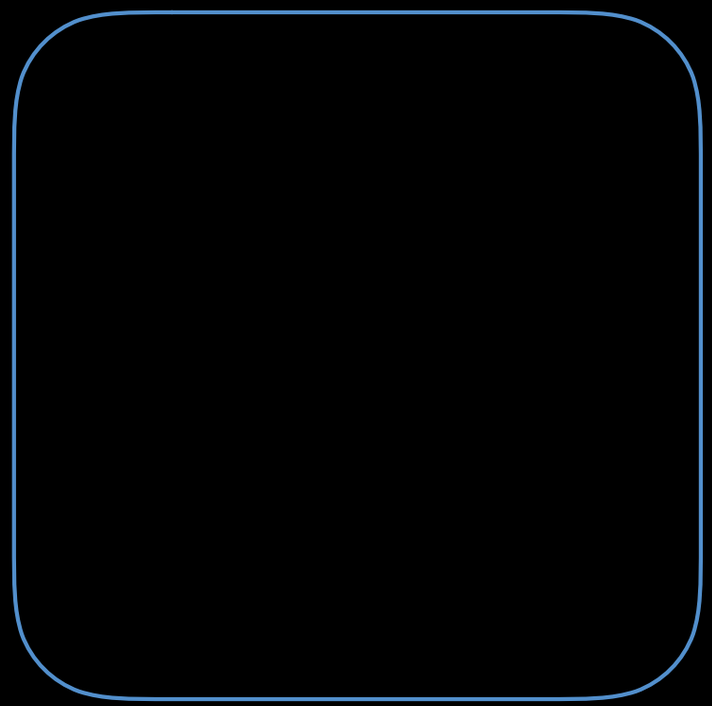
```
["Mohsen": 17, "Amy": 40, "Graham": 5, "Justyn": 67]
```

Optionals

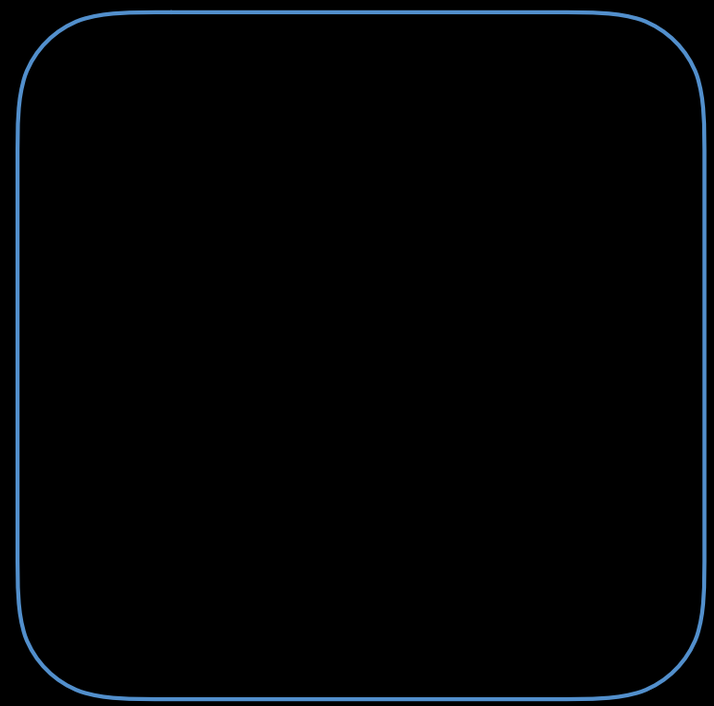
```
let ages = ["Mohsen": 17, "Amy": 40, "Graham": 5]

let possibleAge = ages["Amy"]
```

Optionals

```
ages["Amy"]
```



```
ages["Daryl"]
```



Optionals

`ages["Amy"]`

40

Int

`ages["Daryl"]`

Optionals

`ages["Amy"]`

40

Int

`ages["Daryl"]`

No Value

No Int

Checking for an Optional Value

```
let ages = ["Mohsen": 17, "Amy": 40, "Graham": 5]

let possibleAge: Int? = ages["Daryl"]

if possibleAge == nil {
    print("Age not found.")
}
```

Checking for an Optional Value

```
let ages = ["Mohsen": 17, "Amy": 40, "Graham": 5]

let possibleAge: Int? = ages["Daryl"]

if possibleAge == nil {
    print("Age not found.")
}
```

```
Age not found.
```

If-Let Statement

```
let ages = ["Mohsen": 17, "Amy": 40, "Graham": 5]

if let age = ages["Amy"] {
   print("An age of \(age) was found.")
}
```

If-Let Statement

```
let ages = ["Mohsen": 17, "Amy": 40, "Graham": 5]

if let age = ages["Amy"] {
   print("An age of \(age) was found.")
}
```

```
An age of 40 was found.
```

Functions as Parameters

```
func filterInts(_ numbers: [Int], _ includeNumber: type) -> [Int] {…}
```

Functions as Parameters

```
func filterInts(_ numbers: [Int], _ includeNumber: (Int) -> Bool) -> [Int] {




}
```

Functions as Arguments

```
func filterInts(_ numbers: [Int], _ includeNumber: (Int) -> Bool) -> [Int] {…}

let numbers = [4, 17, 34, 41, 82]
func divisibleByTwo(_ number: Int) -> Bool {
    return number % 2 == 0
}

let evenNumbers = filterInts(numbers, divisibleByTwo)
```

Functions as Arguments

```
func filterInts(_ numbers: [Int], _ includeNumber: (Int) -> Bool) -> [Int] {…}

let numbers = [4, 17, 34, 41, 82]
func divisibleByTwo(_ number: Int) -> Bool {
    return number % 2 == 0
}

let evenNumbers = filterInts(numbers, divisibleByTwo)
print(evenNumbers)
```

```
[4, 34, 82]
```

Closure Expressions

```
func filterInts(_ numbers: [Int], _ includeNumber: (Int) -> Bool) -> [Int] {…}

let numbers = [4, 17, 34, 41, 82]
func divisibleByTwo(_ number: Int) -> Bool {
    return number % 2 == 0
}

let evenNumbers = filterInts(numbers, { (number: Int) -> Bool     return number % 2 == 0 })
print(evenNumbers)
```

```
[4, 34, 82]
```

Type Inference in Closures

```
func filterInts(_ numbers: [Int], _ includeNumber: (Int) -> Bool) -> [Int] {…}

let evenNumbers = filterInts(numbers, { number in return number % 2 == 0 })
print(evenNumbers)
```

```
[4, 34, 82]
```

Type Inference in Closures

```
func filterInts(_ numbers: [Int], _ includeNumber: (Int) -> Bool) -> [Int] {…}

let evenNumbers = filterInts(numbers, { number in number % 2 == 0 })
print(evenNumbers)
```

```
[4, 34, 82]
```

Implicit Arguments in Closures

```
func filterInts(_ numbers: [Int], _ includeNumber: (Int) -> Bool) -> [Int] {…}

let evenNumbers = filterInts(numbers, { $0 % 2 == 0 })
print(evenNumbers)
```

```
[4, 34, 82]
```

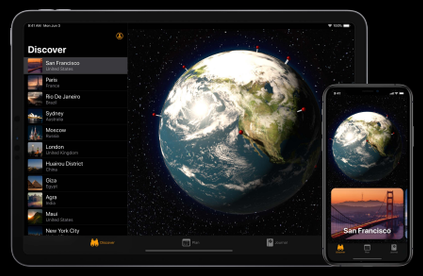
Implicit Arguments in Closures

```
func filterInts(_ numbers: [Int], _ includeNumber: (Int) -> Bool) -> [Int] {…}

let evenNumbers = filterInts(numbers, { $0 % 2 == 0 })
print(evenNumbers)
```

```
[4, 34, 82]
```

The shortest path to building great apps on every device

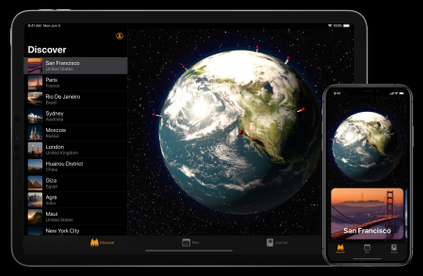
AppKit



UIKit

TVUIKit
UIKit

WatchKit



AppKit

UIKit

TVUIKit
UIKit

WatchKit

Designed for Every Platform

Designed to accommodate many UI paradigms

Platform support out of the box

A common toolkit that you can learn once and use anywhere

Swift Android Compiler

- GCC 4.2

- LLVM-GCC

- LLVM Compiler

How does Swift Android Compiler work?

- Google already makes use of LLVM compilers for native Android development with C/C++. That's why, from an Android device perspective, there is no difference in libraries compiled from C/C++ or Swift code. Moreover, it's actually better for developers, because they can re-use most of the instruments that were created for C/C++ development — Android profiling tools or an Android low-level debugger (with a limitation on an evaluation of Swift code).

What frameworks are available?

- SwiftCore, Dispatch, and SwiftFoundation are the current frameworks available. SwiftCore and Dispatch are largely identical to iOS and Mac versions. But SwiftFoundation on the other hand is not the same Foundation that Apple uses for its own platforms; it's a re-implementation of all classes from the original library. That's why SwiftFoundation is still missing some features, but it covers all basic needs such as performing network requests, parsing JSON/XML, storing data on disk, etc.

Introducing the Swift Android Toolchain

- A collection of tools that gives Android developers the ability to use Swift in their projects comfortably and without any hassles. It is an open-source project based on Apple Swift compiler and SwiftJava and SwiftAndroid projects.

More Information

https://github.com/apple/swift/blob/master/docs/Android.md
https://github.com/readdle/swift-java-codegen
https://github.com/readdle/swift-android-architecture