ANDROID DEVELOPMENT USING KOTLIN

By: Eng. Shymaa Othman

Instructor

- + 10 years experience in Mobile Development.
- Mobile Technical Lead @ Trufla Technology.
- Master Strudent @ Faculty of Gradutate Studies , Computer Science.
- Email : Shymaa.o@pg.cu.edu.eg
- Mobile/whatsapp: 01007762617
- Linkedin: https://www.linkedin.com/in/shymaa-othman-00106029/

Why Learning Android ?

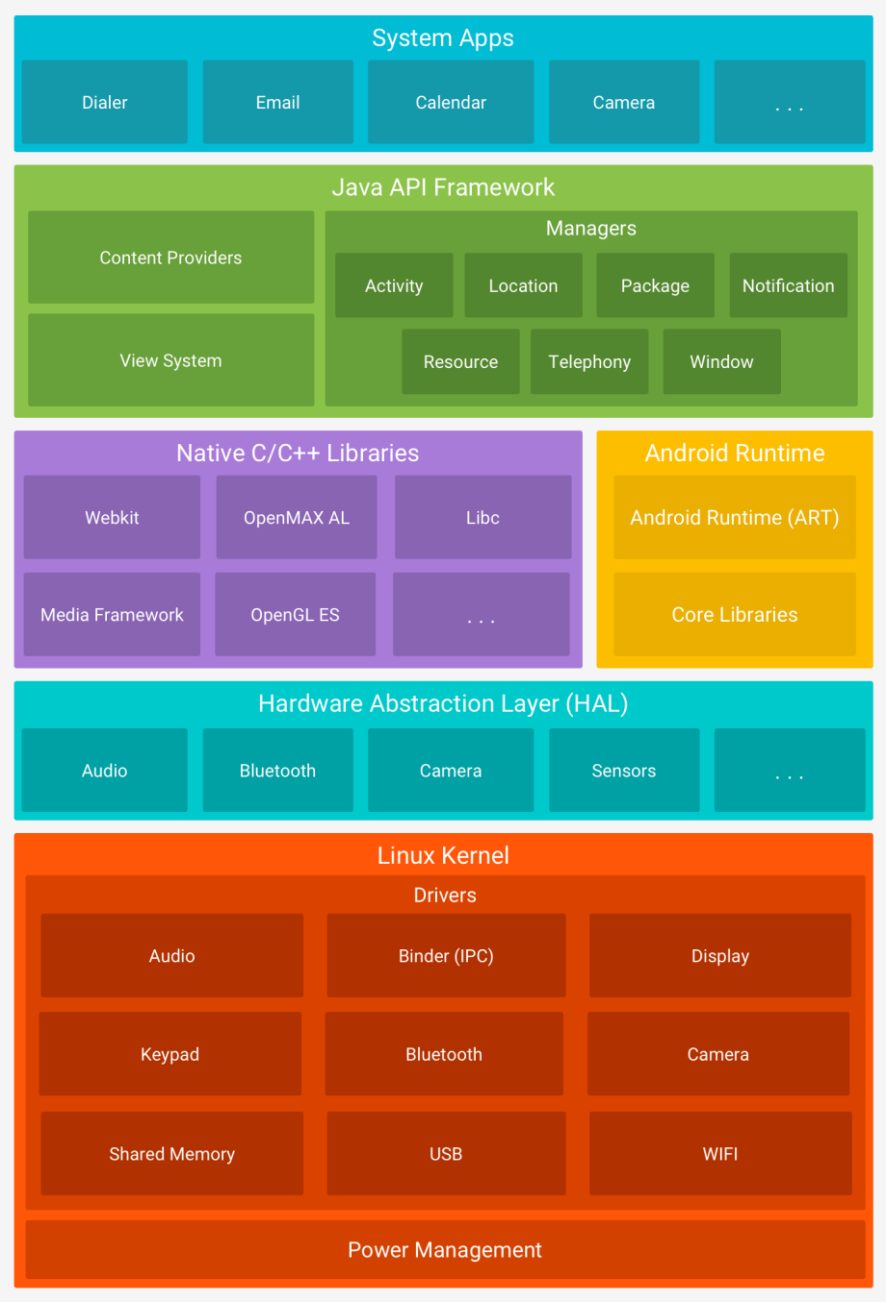
OPEN SOURCE CODE
AVAILABLE EQUIPMENT TO START LEARNING
HUGE COMMUNITY

HUGE DEVICE MARKET SHARE

EASIER ENTRY TO MARKET

Android Operating System

Linux Kernel: threading & low level memory management
HAL: standard interface to expose device hardware capabilities
To higher level JAVA API framework

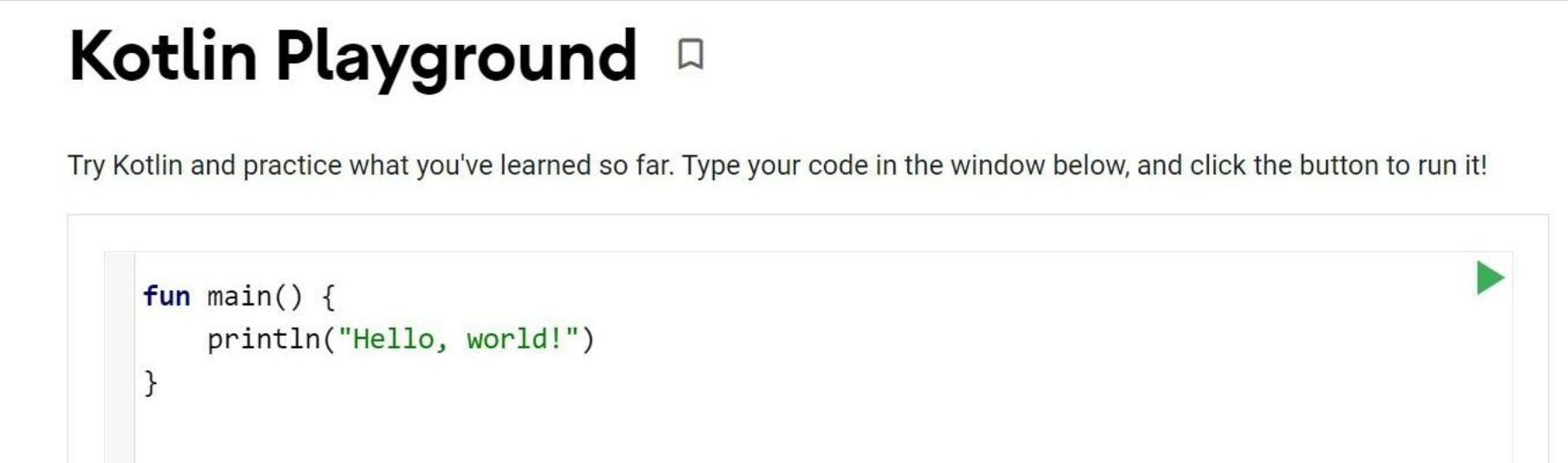


Course Handle

1. Explain Kotlin Lessons.
2. Try simple Kotlin programs online using Kotlin Playground.
3. Apply on Google Code Labs.
4. Solve Quizzes and take Badges.

Write your First Program in Kotlin

Kotlin Playground
https://play.kotlinlang.org/
is an interactive code editor on the web where you can practice writing Kotlin programs.



Lesson1:Kotlin Basics

Program entry point

An entry point of a Kotlin application is the main function.

```
fun main() {
    println("Hello world!")
}
```

Another form of main accepts a variable number of String arguments.

```
fun main(args: Array<String>) {
    println(args.contentToString())
}
```

Lesson1:Kotlin Basics

Print statement

print prints its argument to the standard output.

```
fun main() {
//sampleStart
    print("Hello ")
    print("world!")
//sampleEnd
}
```

println prints its arguments and adds a line break, so that the next thing you print appears on the next line.

```
fun main() {
//sampleStart
    println("Hello world!")
    println(42)
//sampleEnd
}
```

Lesson1:Kotlin Basics

Defining Variables

Val : Read-only local variables are defined using the keyword val. They can be assigned a value only once.

```
fun main() {
//sampleStart
    val a: Int = 1  // immediate assignment
    val b = 2   // `Int` type is inferred
    val c: Int  // Type required when no initializer is provided
    c = 3       // deferred assignment
//sampleEnd
    println("a = $a, b = $b, c = $c")
}
```

Lesson1:Kotlin Basics

Defining Variables

Var: **Variables that can be reassigned** use the var keyword.

```
fun main() {
//sampleStart
    var x = 5 // `Int` type is inferred
    x += 1
//sampleEnd
    println("x = $x")
}
```

Lesson1:Kotlin Basics

Defining Variables

You can declare variables at the top level.

```
//sampleStart
val PI = 3.14
var x = 0

fun incrementX() {
    x += 1
}
//sampleEnd

fun main() {
    println("x = $x; PI = $PI")
    incrementX()
    println("incrementX()")
    println("x = $x; PI = $PI")
}
```

Lesson1:Kotlin Basics

Comments

Comments Just like most modern languages, Kotlin supports single-line (or end-of-line) and multi-line (block) comments.

```
// This is an end-of-line comment

/* This is a block comment
   on multiple lines. */
```

Block comments in Kotlin can be nested.

```
/* The comment starts here
/* contains a nested comment *â /
and ends here. */
```

Lesson1:Kotlin Basics

Defining Functions

A function with two **Int parameters** and **Int return type**.

```kotlin
//sampleStart
fun sum(a: Int, b: Int): Int {
    return a + b
}
//sampleEnd

fun main() {
    print("sum of 3 and 5 is ")
    println(sum(3, 5))
}
```

Lesson1:Kotlin Basics

Defining Functions

A function **body** can be an **expression**. Its **return type is inferred**.

```kotlin
//sampleStart
fun sum(a: Int, b: Int) = a + b
//sampleEnd

fun main() {
    println("sum of 19 and 23 is ${sum(19, 23)}")
}
```

Lesson1:Kotlin Basics

Defining Functions

A function that **returns no meaningful value**.

```
//sampleStart
fun printSum(a: Int, b: Int): Unit {
    println("sum of $a and $b is ${a + b}")
}
//sampleEnd

fun main() {
    printSum(-1, 8)
}
```

Lesson1:Kotlin Basics

Defining Functions

Unit **return type** can be **omitted**.

```
//sampleStart
fun printSum(a: Int, b: Int) {
    println("sum of $a and $b is ${a + b}")
}
//sampleEnd

fun main() {
    printSum(-1, 8)
}
```

Kotlin Basics

repeat Function

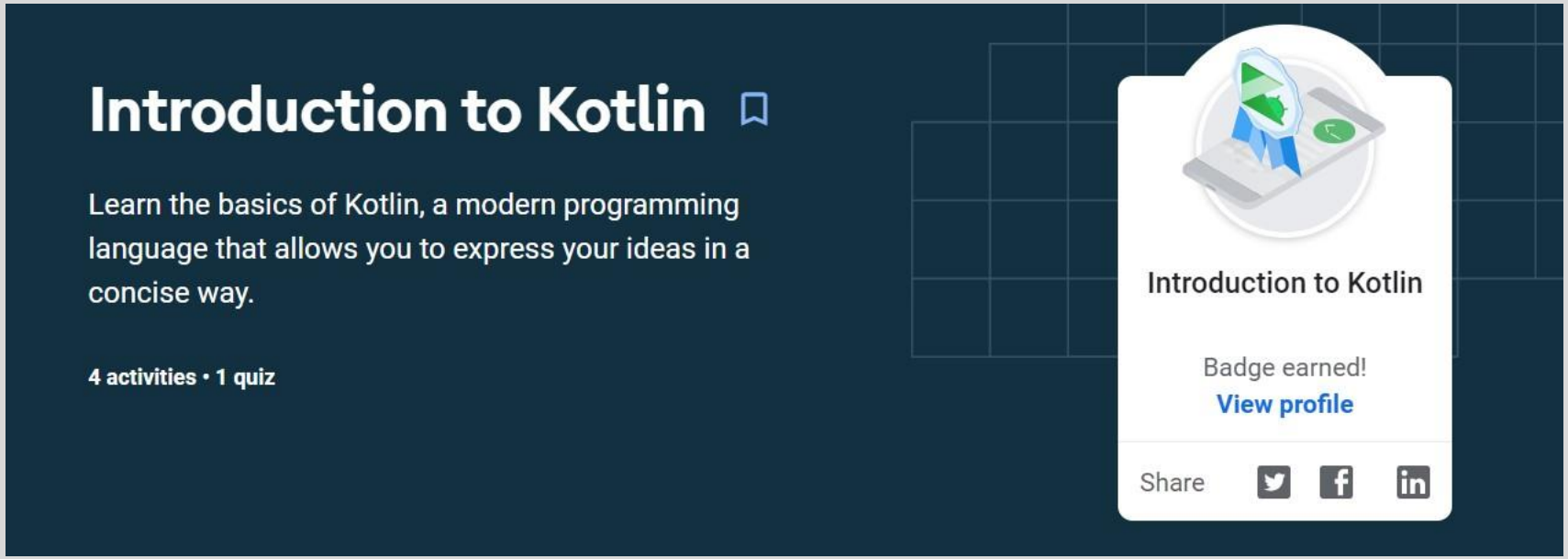
Executes the given function action specified number of times.
A zero-based index of current iteration is passed as a parameter to action.

```
// greets three times
repeat(3) {
    println("Hello")
}

// greets with an index
repeat(3) { index ->
    println("Hello with index $index")
}

repeat(0) {
    error("We should not get here!")
}
```

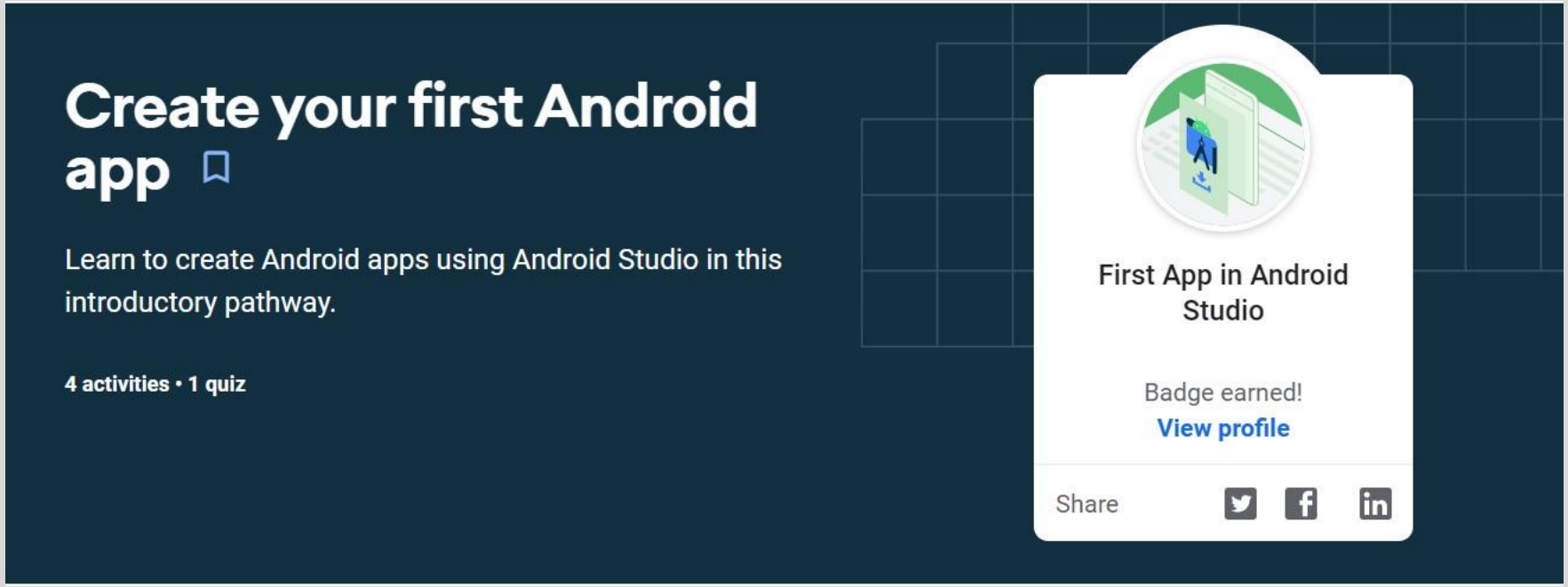
Codelab1 + Quiz1



Lesson2: Create your first Android app

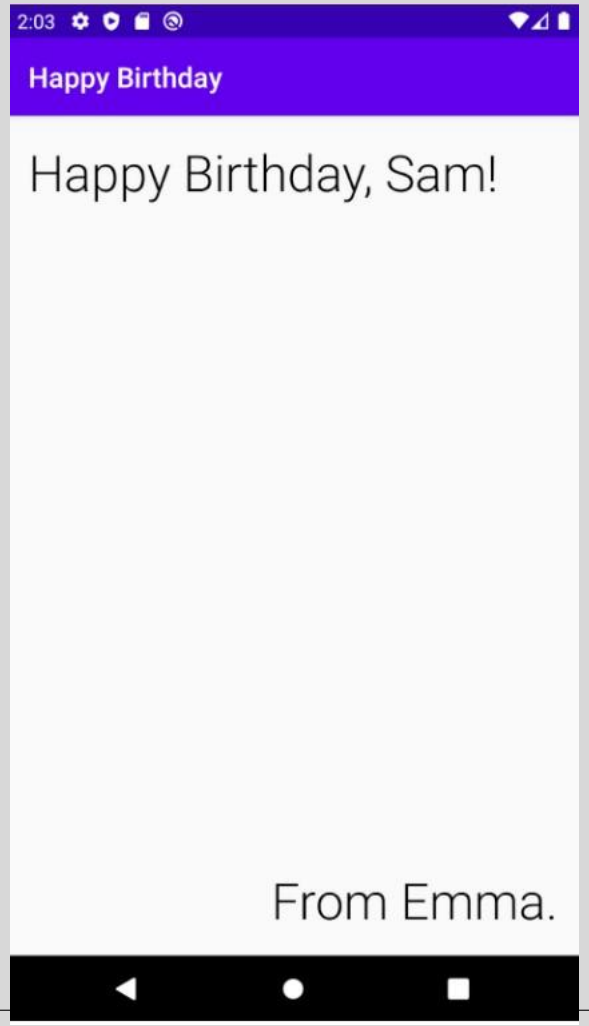
- Introduction to Android Studio.
- Download and Install Android Studio.
- Create and Run your first Android App.
- Run Your App on a Mobile Device.
- Quiz 2

Codelab2 + Quiz2



Lesson2: Create your first Android app

Create a Birthday Card app



- The **Layout Editor** helps you create the UI for your Android app.
- Almost everything you see on the screen of your app is a **View**.
- A **TextView** is a UI element for displaying text in your app.
- A **ConstraintLayout** is a container for other UI elements.
- **Views** need to be constrained horizontally and vertically within a **ConstraintLayout**.
- One way to position a **View** is with a **margin**.
- A **margin** says how far a **View** is from an edge of the container it's in.
- You can set **attributes** on a **TextView** like the font, text size, and color.

Lesson2: Create your first Android app

Add images to your Android app

- The **Resource Manager** in Android Studio helps you add and organize your images and other resources.
- An **ImageView** is a UI element for displaying images in your app.
- **ImageViews** should have a content description to help make your app more accessible.
- **Text** that is shown to the user like the birthday greeting should be **extracted into a string** resource to make it easier to **translate** your app into other languages.

Codelab3 + Quiz3