IEEE Accelerator Program

App Dev
Getting Started with Kotlin

IEEE Computer Society, MUJ

January 2022

Week 2

Agenda

1. What is Kotlin?
2. Java vs Kotlin
3. Kotlin Basics
4. Intro to Android and Android Studio

What is Kotlin?

- Kotlin is a general-purpose programming language which first appeared in 2011 and it is designed to fully interoperate with Java, and the JVM.

- Kotlin mainly targets the JVM, but can also be compiled to JavaScript (e.g., for frontend web applications using React) or native code (via LLVM); e.g., for native iOS apps sharing business logic with Android apps.

- Nowadays Kotlin is famously known to be mainly used while developing Android Apps, but as we already saw that it can be used for other general purpose or specifc operations as well.

Java vs Kotlin

Java

```
import java.util.Scanner;

public class java {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String name = scanner.nextLine();
        System.out.println("Hello " + name);
        scanner.close();
    }
}
```

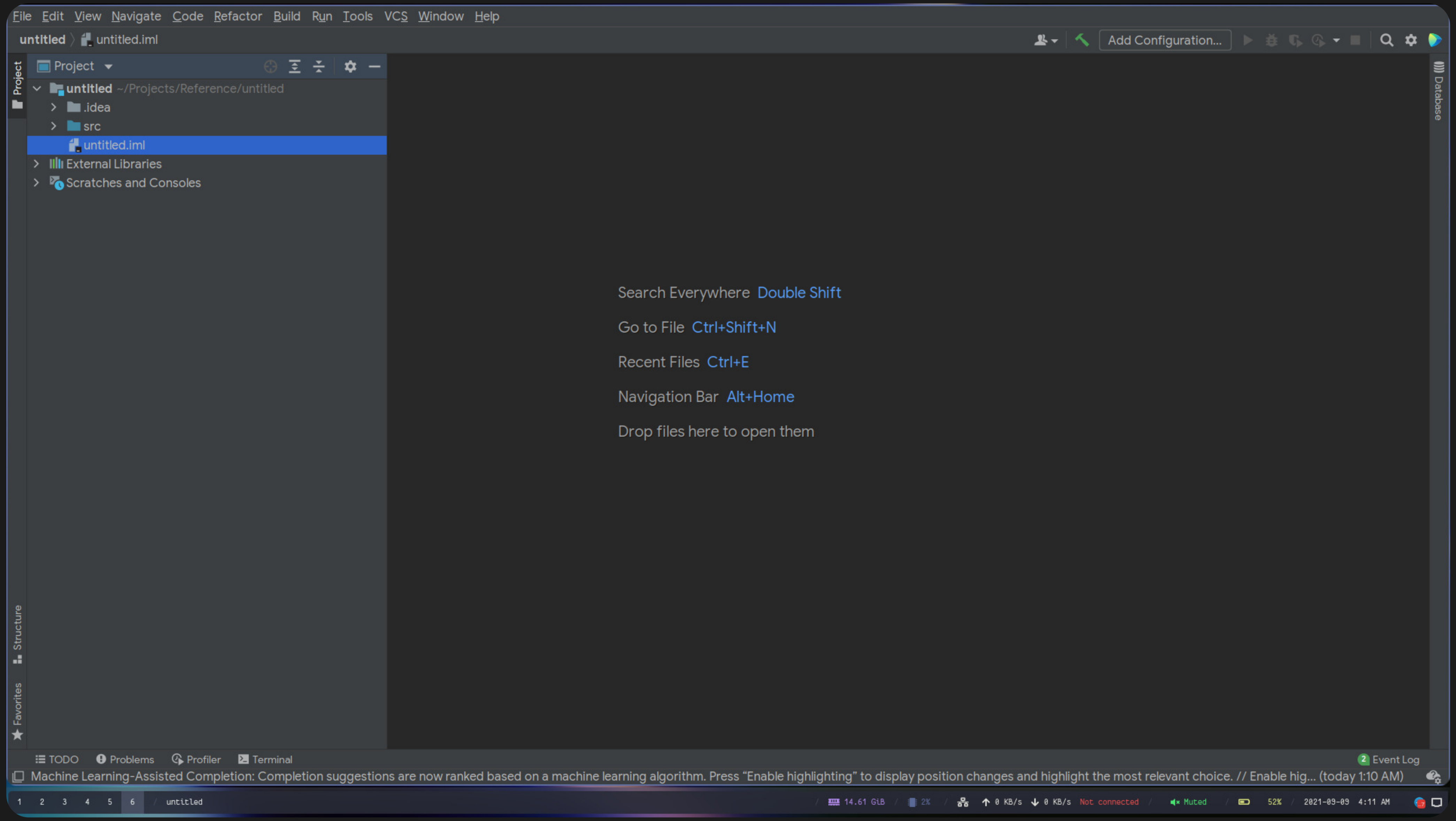
Kotlin

```kotlin
fun main() {
    val name = readLine()
    println("Hello $name")
}
```

How to test Kotlin Code?

1. IntelliJ Idea

- IntelliJ Idea is the official IDE developed by JetBrains.
- Great platform for developing projects using Kotlin and Java.
- https://www.jetbrains.com/idea/



Search Everywhere Double Shift

Go to File Ctrl+Shift+N

Recent Files Ctrl+E

Navigation Bar Alt+Home

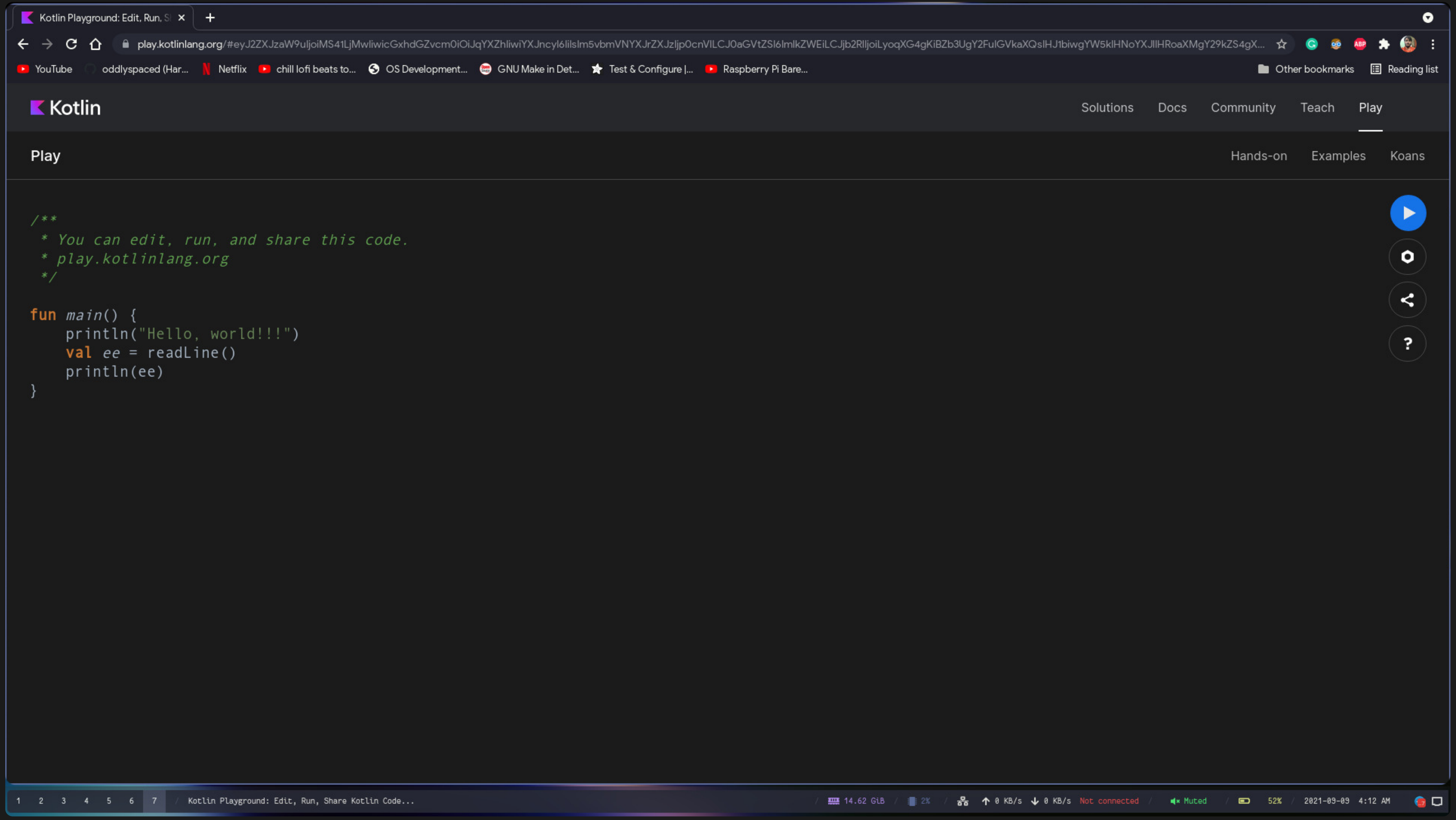
Drop files here to open them

Machine Learning-Assisted Completion: Completion suggestions are now ranked based on a machine learning algorithm. Press “Enable highlighting” to display position changes and highlight the most relevant choice. // Enable hig... (today 1:10 AM)

```
1 2 3 4 5 6 untitled        14.61 GiB  2%  0 KB/s  0 KB/s Not connected  Muted  52%  2021-09-09 4:11 AM
```

2. Kotlin Playground

- Kotlin Playground is an amazing in browser tool to test and learn basic Kotlin Code
- https://play.kotlinlang.org/



```
/**
 * You can edit, run, and share this code.
 * play.kotlinlang.org
 */

fun main() {
    println("Hello, world!!!")
    val ee = readLine()
    println(ee)
}
```

Kotlin Basics

Hello World

- Hello World in Kotlin is very easy, and if you have any programming idea with languages like Java or C++ then the following code would probably look relevant to you.

```
// Hello World
fun main() {
    println("Hello World!")
}
```

Variables

- Variables are used to store data in our programs.
- Kotlin uses two different keywords to declare variables: val and var.
- Use val for a variable whose value never changes. You can't reassign a value to a variable that was declared using val.
- Use var for a variable whose value can

```
var count: Int = 10
count = 15 // Possible

val languageName: String = "Kotlin"
languageName = “Java” // Not Possible
```

Data Types

- Kotlin has the following basic Data Types :
  a. Integer Data type
  b. Floating-point Data Type
  c. Boolean Data Type
  d. Character Data Type

```
var myint = 35 // Int
var mylong = 23L // Long
var myFloat = 54F // Float
val myBool = false // Boolean
val myChar = ‘Z’ // Character
val myString = “Hello” // String
```

Arithmetics

- Arithmetics in Kotlin is very much similar to what we’ve been doing in other languages.

```
var add = 1 + 2 // Add
add += 3
var sub = 5 - 3 // Subtract
sub -= 2
var mul = 4 * 2 // Multiply
mul *= 10
var div = 6 / 3 // Division
div /= 3
var rem = 10 % 4 // Remainder
```

Conditional Statements (if)

- Kotlin has the typical If, Else If, Else statement block for conditions.
- These can be incorporated with assignment operations to make variable assignment simpler.

```
var max = 10
if (max < 15) {
    println(“Small”)
} else if (max == 10) {
    println(“Equal”)
} else {
    println(“Larger”)
}
```

```
val max = if (a > b) {
    print("Choose a")
    a
} else {
    print("Choose b")
    b
}
```

Conditional Statements (when)

- Kotlin has the normal If - Else block, while additionally a When expression for more complex and dynamic operations.

```
when (x) {
    1 -> print("x == 1")
    2 -> print("x == 2")
    else -> { // Note the block
        print("x is neither 1 nor 2")
    }
}
```

Loops (for)

- The for loop iterates through anything that provides an iterator.
- Syntax is a bit different when compared to other languages.

```
// Looping over a range
for (i in 0..10) {
    println(i)
}

// Looping over an array
for (value in someArray) {
     println(value)
}
```

Loops (while)

- while and do-while loops execute their body continuously while their condition is satisfied. The difference between them is the condition checking time.

```
var x = 10
while (x > 0) { // Standard while loop
    x--
}

// Do While
do {
    val y = retrieveData()
} while (y ≠ null) // y is visible here!
```

Functions

- Kotlin functions are declared using the fun keyword

```
// Declaring a function
fun doubleNum(x: Int): Int {
    return 2 * x
}

// Using a function
val result = doubleNum(x: 2)
```

Arrays

- In Kotlin, arrays are not a native data type, but a mutable collection of similar items which are represented by the Array class.

```
// Array Initialisation without explicitly providing a type
val num1 = arrayOf(1, 2, 3, 4)

// Array Initialisation with a type
val num2 = arrayOf<Int>(1, 2, 3)

// Array Initialisation with mixed data type (Type of array
// becomes Array<Object>)
val num3 = arrayOf(1, “hi”, ‘A’)
```

List vs Array List

- Lists are collections and they can be mutable in nature.
- Normal lists are immutable by default, that means that once defined they cannot be changed.
- ArrayList on the other hands are mutable and you can resize them, add, remove items at will.

```
// Immutable List
listOf(1, 2, 3)

// Mutable List
arrayListOf(1, 2, 3)
```

Class

- Kotlin class is similar to Java class, where a class is a blueprint for the objects which have common properties.
- Kotlin classes are declared using the keyword class.
- A class has a class header which specifies its type parameters, constructor etc. and the class body which is surrounded by curly braces.
- A class body contains the multiple functions, additional constructors, companion object and any parameters.

Class Example

```
class Hello (private val a1: Int) {
    init {
        println(“Setting up...”)
    }
    companion object {
        val abc = "111"
    }
    fun test() {
        print(“H!”)
    }
}
```

Public vs Private

- Any function or property tagged with the keyword ‘public’ is accessible by any other class.
- Opposing this, functions and properties tagged with ‘private’ are only accessible inside the class.

```
class Hello {
    private val someValue = 100 // Not usable outside Hello
    public val someAnotherValue = 10 // Accessible anywhere
    val anotherValue = 90 // Same as public
}
```

Fin

Kotlin Resources

Official Kotlin Getting Started Guide:

- https://kotlinlang.org/docs/basic-syntax.html

Alternate Resource (TutorialsPoint)

- https://www.tutorialspoint.com/kotlin/index.htm

Alternate Resource (JavaTPoint)

- https://www.javatpoint.com/kotlin-tutorial

FreeCodeCamp Kotlin Marathon

- https://www.youtube.com/watch?v=F9UC9DY-vIU

Educative Course - Kotlin [Advance, Paid]

- https://www.educative.io/courses/ultimate-guide-programming-in-kotlin

Important :
Program Flow
Variables
Data Types
Scope of variables
Functions
Loops
Classes
Data Class
Sealed Class
Arrays
List
ArrayList
Map
HashMap

@haaaaaaardik

@srivastavahardik

+91 8005493889