Introduction to Kotlin

By Muhammad Haris Bin Abid

What is Kotlin?

Introduction

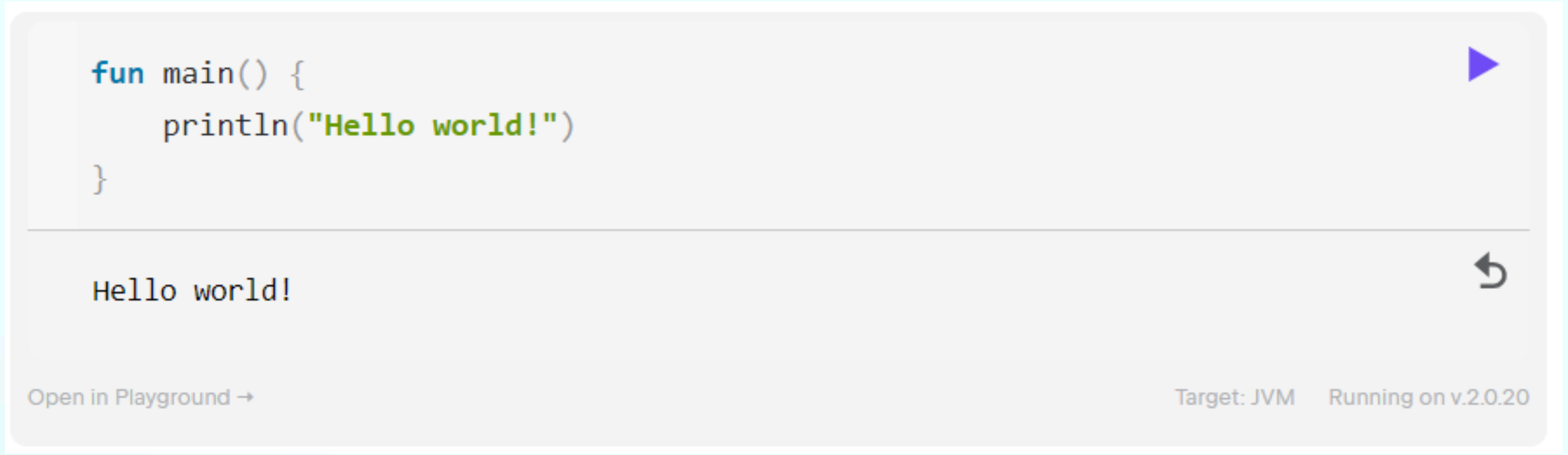
- A modern, statically typed programming language.
- Developed by JetBrains, first released in 2011.
- Officially supported by Google for Android development.
- Fully interoperable with Java (Kotlin Code → Kotlin Compiler → Java Bytecode → JVM → Execution)
- Similar to C++ but designed for safety and conciseness.

Why Use Kotlin?

- **Concise**: Reduces boilerplate code.
- **Safe**: Avoids null pointer exceptions with nullable types.
- **Interoperable**: Works seamlessly with Java.
- **Expressive**: More readable and easier to write.
- **Coroutines**: Simplifies asynchronous programming.
- **Functional Programming**: Supports higher-order functions and lambdas.

Kotlin Entry Point

- The Entry Point of the Kotlin program is *main* Function



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

```
Hello world!
```

-
- ”print” prints its argument to the standard output
- ”println” its arguments and adds a line break, so that the next thing you print appears on the next line
- Use ‘play.kotlinlang.org' to practice online

Kotlin Basics - Data Types

- val byte: Byte = 127 *// 8 bits, -128 to 127*
- val short: Short = 32767 *// 16 bits, -32768 to 32767*
- val int: Int = 2147483647 *// 32 bits, -2^31 to 2^31-1*
- val long: Long = 9223372036854775807 *// 64 bits, -2^63 to 2^63-1 // Floating-point numbers*
- val float: Float = 3.14f *// 32 bits, 6-7 decimal digits*
- val double: Double = 3.14 *// 64 bits, 15-17 decimal digits*
- val char: Char = 'A' *// Single character*
- val string: String = "Hello" *// Text*
- val isTrue: Boolean = true *// Boolean*

Kotlin Basics - Variables

- In Kotlin, you declare a variable starting with a keyword ***val*** or ***var***, followed by the name of the variable
- Val: Used to declare variables that are assigned a value only once (Think about constants)
  - They are immutable
    - val x: Int = 5
- Var: Used to declare variables that are assigned a value multiple times (Now think about variable)
  - They are mutable
    - Var y: Int = 5

Kotlin Basics - Variables

- Kotlin supports type inference and automatically identifies the data type of a declared variable
    - val country = "Pakistan" // Compiler infers String type
- When declaring a variable, you can omit the type after the variable name
- You can use variables only after initializing them
- You can either initialize a variable at the moment of declaration or declare a variable first and initialize it later

Kotlin Basics - Strings

1. String Concatenation

```
val firstName = "John"
val lastName = "Doe"
val fullName = firstName + " " + lastName // "John Doe”
```

2. String Interpolation

```
val name = “John”
 val greeting = "Hello, $name!" // "Hello, John!"
```

Expression interpolation

```
val age = 25
val message = "I'll be ${age + 1} next year"
```

Kotlin Basics - Nullable vs Non-Nullable

Non-nullable (can't be null)

```kotlin
var name: String = "John"

name = null  // Compilation Error!
```

Nullable (can be null)

```kotlin
var name: String? = "John"

name = null  // OK
```

Safe Operators

```kotlin
val name: String? = "John"

val length: Int? = name?.length // Safe access
```

Kotlin Basics - Nullable vs Non-Nullable

Elvis Operator (?:)

```
val name: String? = null
val length: Int = name?.length ?: 0 // Default value if null
```

Not-null Assertion (!!)

```
val name: String? = "John"
val length: Int = name!!.length // Throws NPE if null
```

Kotlin Basics - functions

- **Definition :** fun functionName(parameter1: Type1, parameter2: Type2): ReturnType { // *function body* return result }

- **Single line :** fun sum(a: Int, b: Int) = a + b

- **Default Parameters :** fun displayMessage(msg: String = "Welcome!") {

  println(msg)

}

Kotlin Basics - Control Flow

If condition

```
val number = 10

if (number > 0) {

    println("Positive")

} else {

    println("Negative")

}
```

When expression (like switch-case in C++)

```
val day = 3

val result = when(day) {

    1 -> "Monday"

    2 -> "Tuesday"

    3 -> "Wednesday"

    else -> "Invalid day"

}
```

Kotlin Basics - Loops

For Loop

```
for (i in 1..5) {
    println(i) // Prints 1, 2, 3, 4, 5
}
```

Collection Loop

```
val fruits = listOf("apple", "banana", "orange")
for (fruit in fruits) {
    println(fruit)
}
```

With Index

```
for ((index, fruit) in fruits.withIndex()) {
    println("$index: $fruit")
}
```

Kotlin Basics - Loops

While vs Do-While

while loop

```
var count = 0 while (count < 5) {
    println(count)
    count++
}
```

Do-While

```
var number = 1
do {
 println(number) number++
 } while (number <= 3)
```

Kotlin Basics - Loops

Different ways to loop through ranges

- for (i in 1..5) // 1 to 5
- for (i in 1 until 5) // 1 to 4
- for (i in 5 downTo 1) // 5 to 1
- for (i in 0..10 step 2) // 0, 2, 4, 6, 8, 10

Kotlin Basics - Collection

1. List

- **Ordered collection**
- **Can contain duplicates**

```kotlin
// Immutable List val readOnlyList = listOf("apple", "banana", "orange")

// Mutable List val mutableList = mutableListOf("apple", "banana", "orange")

mutableList.add("grape")
```

2. Set

- **Unique elements only**
- **No duplicates allowed**

```kotlin
// Immutable Set val readOnlySet = setOf("apple", "banana", "orange")

// Mutable Set val mutableSet = mutableSetOf("apple", "banana", "orange")

mutableSet.add("apple") // Won't add duplicate
```

Kotlin Basics - Collection

3. Map

- **Key-value pairs**
- **Keys must be unique**

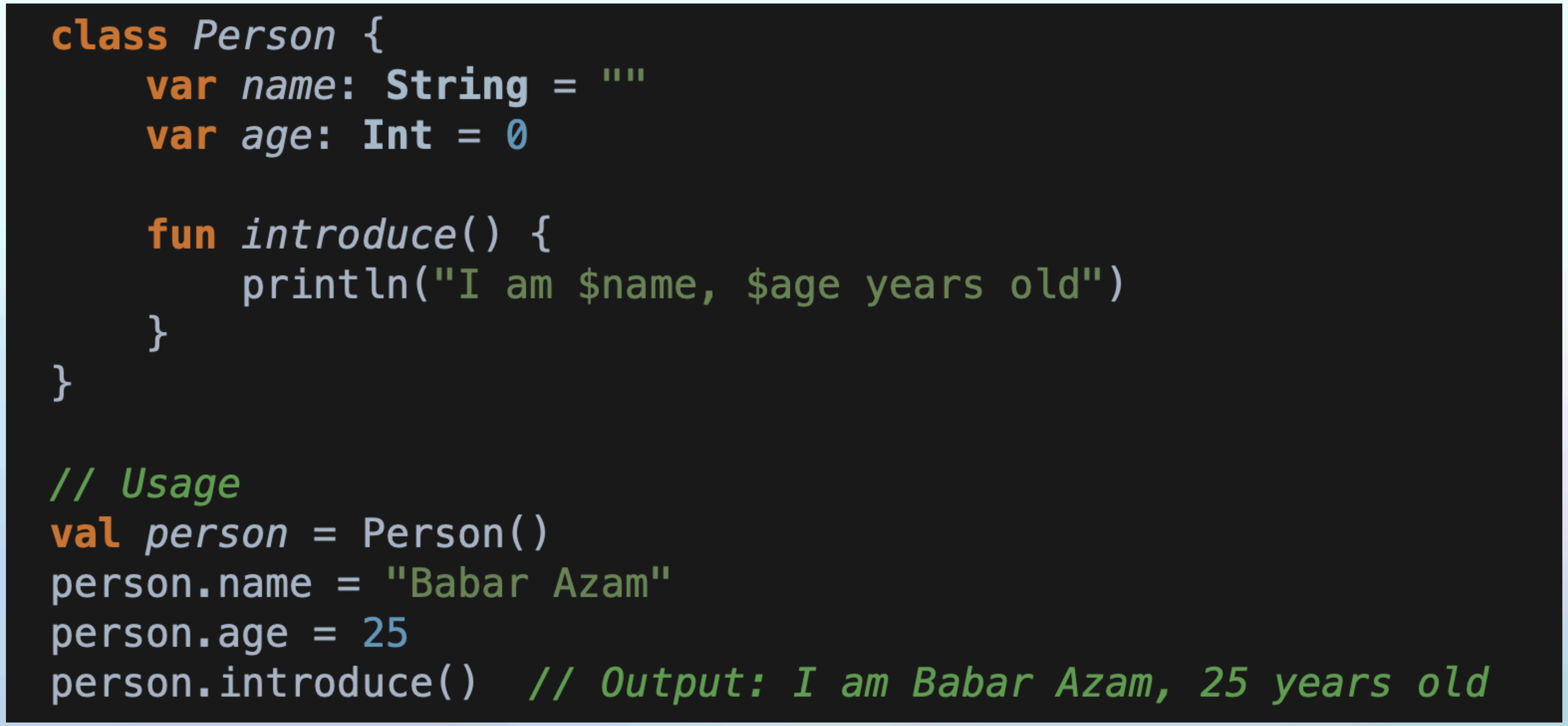
```
// Immutable Map val readOnlyMap = mapOf( "a" to 1, "b" to 2, "c" to 3 ) // Mutable Map

val mutableMap = mutableMapOf( "a" to 1, "b" to 2 )

mutableMap["c"] = 3 // Adding new entry
```

Kotlin Basics - Class

A class is a blueprint or template for creating objects. It encapsulates:

1. Data (properties/attributes)

2. Behaviors (functions/methods)



```kotlin
class Person {
    var name: String = ""
    var age: Int = 0

    fun introduce() {
        println("I am $name, $age years old")
    }
}

// Usage
val person = Person()
person.name = "Babar Azam"
person.age = 25
person.introduce()  // Output: I am Babar Azam, 25 years old
```

Kotlin Basics - Class

A **primary constructor** is a part of the class header. It's the main constructor declared in the class declaration line itself. It initializes the class instance and its properties.

```kotlin
class Student(val name: String, var age: Int) {
    fun study() = println("$name is studying")
}

// Usage
val student = Student("Alice", 20)
```

Thank You!