

# Bootcamp - Android

Roshka  
Agosto - 2022

# Kotlin historia

Kotlin es un lenguaje de programación **compilado** creado en 2010 por JetBrains, la empresa responsable de los IDE IntelliJ.

La razón principal de crearlo fue solucionar ciertos problemas de Java e implementar ciertas funcionalidades que Java no tenía.


Kotlin es un lenguaje que se compila a **bytecode y se ejecuta en la JVM**, esto quiere decir que el código compilado se ejecuta en el mismo ambiente que Java y de hecho es interoperable con Java.

# Ventajas de Kotlin

- Ahorros de getter, setter y otros métodos
- Casteo automático
- Organización de paquetes
- Null safety
- Funciones de alto nivel
- [Documentación](#)

## Lo básico

Como en otros lenguajes como C, Java, toda aplicación de Kotlin va empezar en su función **main** definida de la siguiente forma.




```
fun main() {  
    print("Buenas")  
}
```

# Variables

Las variables las podemos definir con las palabras clave **var** y **val**.

**var** utilizamos para definir variables (que puede cambiar su valor) y **val** para definir valores sólo lectura o constantes.



```
val a: Int = 1 // asignacion inmediata
var b = 2      // tipo `Int` inferido
val c: Int    // el tipo es necesario al no inicializar
c = 3         // asignación diferida
b = 9         // reasignamos valor cuando es var
```

# Tipos de datos

Los tipos de datos pueden ser **enteros**, **decimales**, **booleanos**, **char** y **string**. Pero podemos ver más acerca de estos en su [documentación](#).

Lo bueno de Kotlin es que los tipos de datos están *inferidos* y no hace falta que estemos definiendo explícitamente.


# Operadores

También cada tipo de dato tiene sus operadores específicos, la mayoría de ellos son iguales que otros lenguajes de programación.

Como los numéricos (aritméticos), booleanos (lógicos) y cadenas (concatenación).

# String templates

Una función muy útil que tiene Kotlin para el manejo de cadenas es el string **template**. Que nos permite evaluar variables y concatenarlas de manera más sencilla en las cadenas.



```
val a = "Text1"
val b = "Text2"
val c = "Text3"
// sin template
print("Quiero concatenar mis variables "+a+", "+b+", "+c)
//con template
print("Quiero concatenar mis variables $a, $b, $c")
```