kotlin

_____

HISTORIA

_____

Su nombre proviene del nombre de una isla.

-->Fue creado en 2010 por JetBrains, que es la empresa detrás de IntelliJ IDEA, uno de los mejores IDE de desarrollo para Java.

-->Su mejor opción era Scala, pero lo descartaron por cuestiones de eficiencia y por ser demasiado potente o pesado para la solución que buscaban.

-->Una vez descartado Java, no querían salir del mundo de JVM y decidieron que la mejor opción era crear su propia versión mejorada de Java: Kotlin.

-->Inicialmente fue creado para aplicaciones de escritorio (que era el mercado de JetBrains), pero ahora mismo es el multiplataforma más potente, además del lenguaje oficial de Android.

_____

VARIABLES

_____

SOLO LECTURA

Una variable de solo lectura (read-only) es una variable que no puede ser reasignada.

Para declararlas, usa la palabra reservada val. y especifica su tipo de dato a su derecha con dos puntos (:).

O declárala con su tipo y asígnala en una línea futura.

|_____
_____|

```
|val xPos: Int = 1 // Asignación junto a declaración
|val yPos: Int   // Declaración
|yPos = 5    // Asignación
```

|_____
_____|

El valor que les asignes en su declaración no podrás cambiarlo posteriormente. Si lo intentas el compilador te mostrará este error.

```
_____
|Val cannot be reassigned
|_____
```

MUTABLES

Este tipo de variables las declaras con la palabra reservada var. Te permiten modificar su valor cuando lo desees.

Si haces que la variable del ejemplo de la sección anterior sea mutable, ya no tendrás el error y se asignara el nuevo valor.

```
|_____
|un main() {
| // 2. Variables mutables
|var xPos: Int = 1
|val yPos: Int = 5
|xPos = 2
|println("Coordenada actual del jugador: ($xPos, $yPos)")
|}
|_____
```

Si corres la aplicación obtendrás como salida:

```
Coordenada actual del jugador: (2, 5)
```

INFERENCIA DE TIPO

En Kotlin puedes omitir los tipos de las variables en su declaración, ya que el compilador puede inferirlos de acuerdo al valor de su asignación.

```
|_____
|val playerName = "Oliver" // Se infiere :String
|val playerHealth = 75 // Se infiere :Int
```

|val playerLucky = 0.2 // Se infiere :Doublé

|_____

Pero que el compilador te apoye con la asignación del tipo, no significa que nunca los especifiques o que no respetes el contenido.

Por ejemplo, intentar asignar un tipo de naturaleza distinta.

|_____

|var playerSpeed = 12

|playerSpeed = "Lento" // Error

|_____

El compilador arrojará el siguiente error, ya que estas asignando una cadena a lo que el compilador había inferido como entero:

Type mismatch: inferred type is String but Int was expected

_____

TIPOS PRIMITIVOS

_____

NUMEROS ENTEROS

Tipo	Tamaño en bits	Límite inferior	Límite superior
Byte	8	-128	127
Short	16	-32768	32767
Int	32	-2-31	231-1
Long	64	-263	263-1

NUMEROS REALES

Tipo	Tamaño en bits	Bits significativos	Bits del exponente	Dígitos decimales
Float	32	24	8	6-7
Double	64	53	11	15-16

CARACTERES

SE hace con CHAR

Caracteres de escape

\t: Tabulación

\b: Retroceso

\r: Retorno de carro

\n: Salto de línea

\': Apostrofe

\": Comilla doble

\\: Backslash

\$: Símbolo de dólar

\u+XXXX: Símbolo Unicode con 4 dígitos hexadecimales

BOOlEANOS

En el caso de los valores booleanos, usa el tipo Boolean. Tendrás las palabras clave true para verdadero y false para representar falso.

val globalMapEnable: Boolean = true

_____

CONTROL DE FLUJO

_____

```
// if-else
if (x > 0) { … } else { … }


// when
when (valor) {
 1 -> println("Uno")
2, 3 -> println("Dos o Tres")
in 4..10 -> println("Entre 4 y 10")
else -> println("Otro")
}
```

```
// Bucles
for (i in 1..5) { println(i) }
for (item in lista) { println(item) }
while (cond) { … }
do { … } while(cond)
```