Null

var a: String = "abc" // La variable *a* puede contener un String y no puede contener null

var b: String? = "abc" // La variable *b* puede contener un String o puede contener null

b?.lenght // Para acceder a las propiedades de *b* usaremos una llamada segura

b?.contains(‘a’) // Para llamar a las funciones miembro de *b* usaremos una llamada segura

El operador Elvis

var a: String = "abc"

var b: String? = "abc"

a = b ?: “” // Asignamos a *a* el valor de *b* a no ser que esta sea null, en tal caso le asignamos un String vacío

El operador !!

// Intentaremos no usarlo nunca

```
val longitud = b!!.length // Si b es null lanzará NullPointerException
```

Funciones de ámbito

Let

```
val str: String? = "Hello"
val length = str?.let {
    println(it)        // Solo se ejecuta si str no es null
}
```

With

```
val numeros = mutableListOf("uno", "dos", "tres")
with(numeros) {
    println("'with' es llamado con el argumento $this")
    println("Contiene $size elementos")
}
```

Run

```
val servicio = MultiportService("https://example.kotlinlang.org", 80)
val resultado = servicio.run {
    port = 8080
    query(prepareRequest() + " to port $port")
}
```

Apply

```
val adam = Persona("Adam").apply {
    edad = 32
    ciudad = "Madrid"
}
println(adam)
```

Also

```
val numeros = mutableListOf("uno", "dos", "tres")
numeros
    .also { println("La lista de elementos hasta ahora: $it") }
    .add("cuatro")
```

Take-if y take-unless

```
val numero = Random.nextInt(100)
val parONull = numero.takeIf { it % 2 == 0 }
val imparONull = numero.takeUnless { it % 2 == 0 }
```