Menggunakan nullability di Kotlin

Kotlin menangani null dengan tipe yang aman, yang membedakan antara tipe nullable dan non-nullable secara eksplisit.

1. Tipe Nullable dan Non-Nullable di Kotlin

Di Kotlin, secara default, tipe data tidak bisa bernilai `null`. Jika Anda mencoba memberi nilai `null` ke sebuah variabel non-nullable, Anda akan mendapatkan kesalahan kompilasi.

Non-Nullable

```
var nonNullableString: String = "Hello"
// nonNullableString = null // Ini akan menghasilkan kesalahan
```

Nullable

Untuk membuat variabel yang bisa bernilai `null`, tambahkan tanda tanya `?` setelah tipe datanya.

```
var nullableString: String? = "Hello"
nullableString = null // Ini diizinkan
```

Contoh

```
fun main() {
    var nonNullable: String = "Kotlin"
    // nonNullable = null // Ini akan menyebabkan error

    var nullable: String? = "Kotlin"
    nullable = null // Ini diperbolehkan
}
```

---

2. Mengakses Tipe Nullable dengan Safe Call (?.)

Jika Anda ingin mengakses nilai dari variabel nullable, Anda tidak bisa langsung mengakses properti atau metode dari variabel tersebut. Sebaliknya, Anda perlu menggunakan **safe call** (`?.`) untuk menghindari `NullPointerException`.

Contoh

```
fun main() {
    var nullableString: String? = "Hello"

    // Safe call untuk mengakses panjang string
    println(nullableString?.length) // Output: 5

    nullableString = null
    println(nullableString?.length) // Output: null
```

```
}
```

Dalam contoh di atas, `nullableString?.length` akan mengembalikan `null` jika `nullableString` adalah `null`, bukan menghasilkan error.

---

3. Elvis Operator (?:)

Kotlin juga menyediakan **Elvis operator** (`?:`), yang memungkinkan Anda memberikan nilai default jika variabel nullable adalah `null`.

Contoh

```
fun main() {
    var nullableString: String? = null

    // Elvis operator untuk memberi nilai default
    val length = nullableString?.length ?: -1
    println("Panjang string adalah $length") // Output: -1
}
```

Pada contoh di atas, jika `nullableString` bernilai `null`, maka nilai `-1` akan diberikan ke variabel `length`.

---

4. Menggunakan let untuk Mengeksekusi Kode Hanya jika Nilai Tidak null

Kotlin memungkinkan Anda menggunakan fungsi `let` untuk mengeksekusi blok kode hanya jika variabel nullable tidak `null`. Ini sering dipadukan dengan **safe call** (`?.`).

Contoh

```
fun main() {
    var nullableString: String? = "Hello"

    // Menggunakan let untuk eksekusi kode jika tidak null
    nullableString?.let {
        println("Panjang string adalah ${it.length}")
    }

    nullableString = null
    nullableString?.let {
        println("Ini tidak akan dieksekusi karena null")
    }
}
```
