

TÌM HIỂU VỀ NULLABLE

I. Thế nào là Nullable

- Nullable là khái niệm chỉ kiểu dữ liệu cho phép biến có thể nhận giá trị null.
- Trong Kotlin, mặc định mọi biến đều KHÔNG nullable.
Muốn cho phép null, ta bắt buộc phải thêm dấu ? sau kiểu dữ liệu.
- Ví dụ

```
var name: String = "Tra"  
// name = null -> lỗi  
var email: String? = null // nullable
```

II. Khi nào nên/ không nên dùng

1. Nên dùng khi

- Dữ liệu chưa có ngay
- Giá trị không bắt buộc
- Dữ liệu từ API/ Database
- Người dùng chưa nhập

2. Không nên khi

- Biến bắt buộc phải có
- Dùng thường xuyên trong logic
- Muốn code gọn, không check null nhiều

III. Cách thức xử lý null phổ biến như: ?, ?., ?:, let, !!

1. Dấu? – khai báo Nullable

- Dấu? dùng để khai báo một biến có thể nhận giá trị null
- Ví dụ:

```
var name: String?  
name = null  
Biến name có thể có hoặc không có giá trị.
```

2. Toán tử ?. – Safe Call (gọi an toàn)

- Toán tử ?. dùng để truy cập thuộc tính hoặc hàm của biến nullable mà không gây lỗi.
Nếu biến là null thì câu lệnh sẽ không thực hiện và trả về null.
- Ví dụ

```
var name: String? = null  
println (name?.length)
```

Không bị crash dù name là null

3. Toán tử ?: – Elvis Operator (giá trị mặc định)

- Toán tử ?: dùng để gán giá trị mặc định khi biến nullable bằng null.
- Ví dụ

```
var name: String? = null
val result = name ?: "Chưa có tên"
println(result)
Nếu name là null thì result = "Chưa có tên"
```

4. Hàm let – Thực thi khi không null

- let cho phép thực hiện một khối lệnh chỉ khi biến khác null.
- Ví dụ

```
var name: String? = "Tra"

name?.let {
    println("Tên là: $it")
}
Code trong let chỉ chạy khi name không null.
```

5. Toán tử !! – Cam kết không null (nguy hiểm)

- Toán tử !! dùng khi lập trình viên chắc chắn biến không null. Nếu biến thực tế là null → chương trình sẽ crash.
- Ví dụ

```
var name: String? = null
println(name!!.length)
Chạy là lỗi NullPointerException.
```