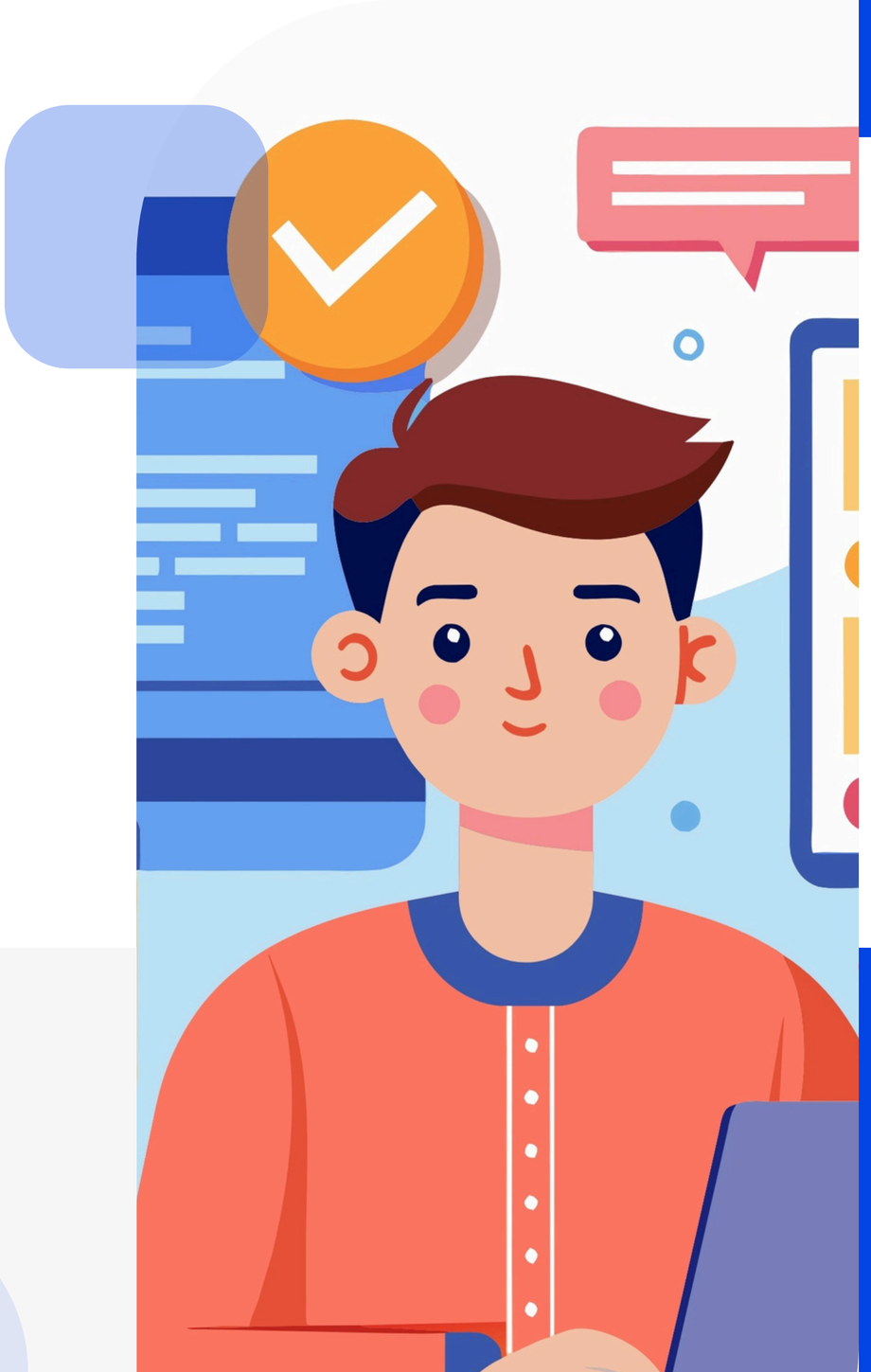
Kotlin scratch project programmer

ARRAYS, COLLECTIONS, AND CLASSES

Giảng Viên hướng dẫn: Trương Quang Tuấn

Trình bày: Phan Hoàng Hiệp



1.GIỚI THIỆU

Lý do chọn đề tài

01 **Tính nền tảng của nội dung**

Trình bày các khái niệm cốt lõi của Kotlin như lập trình hướng đối tượng (OOP), mảng và collections - nền tảng giúp người học nắm vững cấu trúc và tư duy ngôn ngữ trước khi học nâng cao.

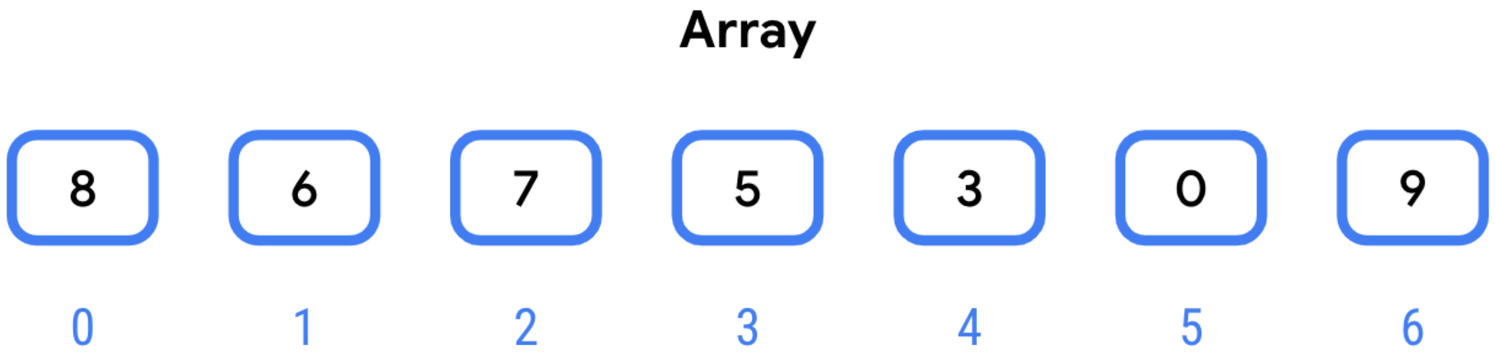
02 **Tính hiện đại và tối ưu trong thiết kế ngôn ngữ Kotlin**

Kotlin được xây dựng với mục tiêu tạo ra một ngôn ngữ ngắn gọn, an toàn và hiệu quả, hướng đến việc giảm thiểu lỗi lập trình và tối ưu hóa năng suất phát triển phần mềm.

03 **Tính ứng dụng và khả năng minh họa thực tiễn**

Các kiến thức trên là nền tảng cho việc xây dựng, tổ chức và quản lý dữ liệu, giúp người học áp dụng hiệu quả vào mọi loại ứng dụng — từ chương trình đơn giản đến hệ thống quy mô lớn.

2. ARRAY



Cú pháp

```
// syntax
//val arrayName = arrayOf(element1, element2, ...)

// example
val fruits = arrayOf("Apple", "Banana", "Mango")

// Any Array
val anyArr: Array<Any> = arrayOf(1, "Hi", 3.14, true)
val mix = arrayOf(1, "Hello", false)

//objects stored in arrays
val people: Array<Person> = arrayOf(
    Person( name: "Robert",  age: 20)
)
```

Lưu ý: Khi sử dụng mảng để lưu đối tượng cần khởi tạo đối tượng trước khi lưu

```
data class Person(val name: String, val age: Int)

val people: Array<Person> = arrayOf(
    Person( name: "Robert",  age: 20)
) // Array<Person> -- Hợp logic

val wrongArray = arrayOf("Brian", 20)
//  Array<Any>  -- Sai logic
```

2. ARRAY

Primitive Arrays

Mảng nguyên thủy (Primitive Array) là mảng lưu trữ trực tiếp các giá trị của kiểu dữ liệu cơ bản.

VD: intArrayOf, doubleArrayOf, charArrayOf,...

```
val numbers = intArrayOf(10, 20, 30)
```

Lưu ý: Kotlin không hỗ trợ mảng nguyên thủy string.

The Array Constructor

Cú pháp: Array(size) { index -> biểu_thức_khởi_tạo }

Ví dụ:

```
val num = Array( size: 5) { i -> i * i }
```

Array Operations

Là các hành động mà lập trình viên có thể thao tác với mảng

Operation	Description	Example
Access	Retrieve an element by its index.	`val element = array[index]`
Update	Modify an element at a specific index.	`array[index] = newValue`
Size	Get the number of elements in the array.	`val size = array.size`
Iterate	Loop through each element in the array.	`for (element in array) {/* ... */}`
Search/find	Check if an element exists in the array (`true` or `false`).	`val found = array.contains(element)`
Slice	Extract a portion of the array.	`val subArray = array.slice(startIndex..endIndex)`
Sort	Arrange elements in ascending or descending order.	`array.sort()` or `array.sortDescending()`
Filter	Create a new array with elements that meet a condition.	`val filteredArray = array.filter {/* condition */}`
Map/transform	Apply a function to each element and create a new array with the results.	`val mappedArray = array.map {/* transformation */}`
Join	Combine elements into a single string with a delimiter.	`val joinedString = array .joinToString(", ")`

2. ARRAY

Multidimensional Arrays

Mảng đa chiều là mảng mà mỗi phần tử của nó lại là một mảng khác.

```
val matrix = Array( size: 3) { row -> // số hàng
    Array( size: 3) { col -> // số cột
        row * col  // giá trị tại mỗi ô = row * col
    }
}
```

	C1	C2	C3
R1	**0**	**0**	**0**
R2	0	1	2
R3	0	2	4

Ta hoàn toàn có thể mở rộng thành mảng 3 chiều bằng cách lồng thêm một Array() nữa. Khi đó, ta sẽ có một mảng chứa mảng, mà mỗi phần tử lại chứa thêm một mảng con khác.

Ứng dụng:

- Ma trận & phép toán số học
- Đồ họa 2D/3D, game
- Xử lý ảnh & video
- Khoa học dữ liệu

3.COLLECTIONS

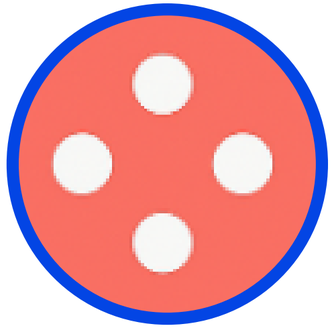
Trong Kotlin, các collection như List, Set và Map chia làm hai loại:

- loại bất biến (immutable) được tạo bằng listOf(), setOf(), mapOf() – không thể thay đổi nội dung sau khi tạo
- loại có thể thay đổi (mutable) được tạo bằng mutableListOf(), mutableSetOf(), mutableMapOf() – có thể thêm, xóa, sửa phần tử.

List

List là tập hợp các phần tử có thứ tự và cho phép trùng lặp.

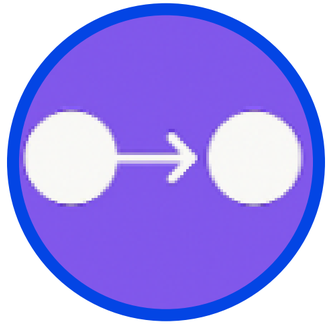
Các hàm hỗ trợ List: first() / last(), indexOf(x), subList(from, to), add(x), remove(x), removeAt(i), clear(), isEmpty(),...



Set

Set là tập hợp không có phần tử trùng nhau, không đảm bảo thứ tự.

Các hàm hỗ trợ Set: union(otherSet), contains(x), add(x), remove(x),...

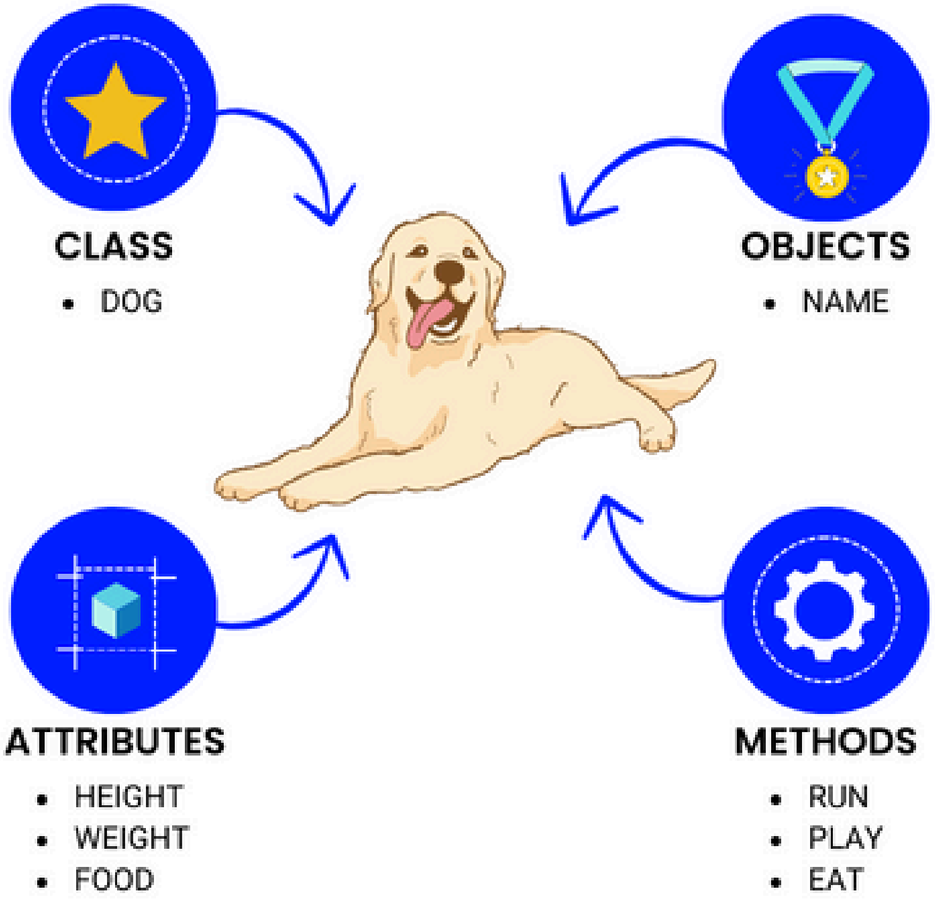


Map

Map là tập hợp của các cặp key–value (key là suy nhất, value có thể trùng).

Các hàm hỗ trợ Map: get(key), put(key, value), remove(key), containsKey(k),...

OBJECT ORIENTED PROGRAMMING



4.OOP

1 Trừu Tượng

Trừu tượng hóa giúp định nghĩa hành vi chung mà chưa cần biết cách thực hiện cụ thể

2 Đóng gói

Đóng gói là che giấu dữ liệu bên trong lớp, chỉ cho phép truy cập thông qua các phương thức được phép

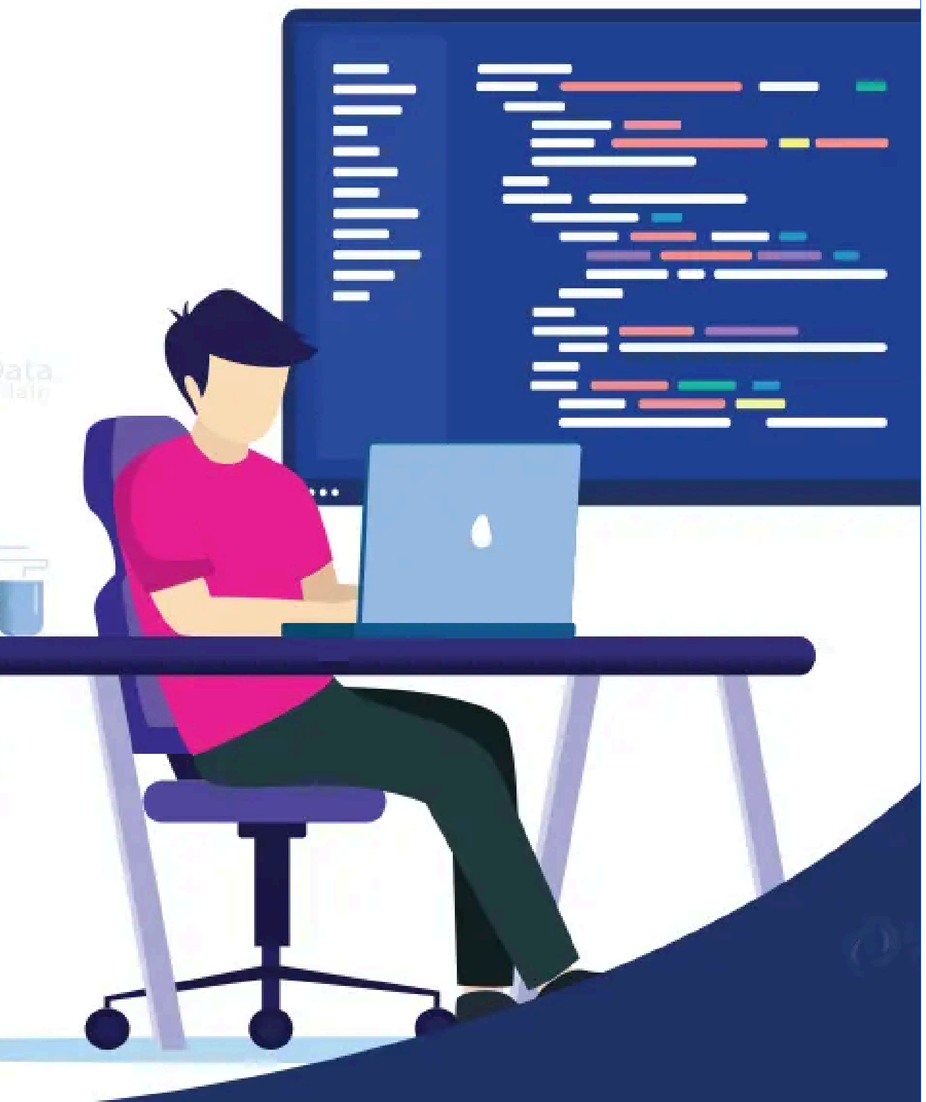
3 Kế Thừa

Kế thừa cho phép một lớp sử dụng lại thuộc tính và hành vi của lớp khác.

4 Đa hình

Đa hình cho phép nhiều lớp con cùng chia sẻ tên phương thức, nhưng hành vi có thể khác nhau.

4.OOP



Abstract & Interface

Tiêu chí	Abstract Class	Interface
Mục đích	Dùng làm lớp cha chung có thể chứa cả thuộc tính & hành vi mặc định.	Dùng để định nghĩa hành vi (contract) mà lớp khác phải tuân theo.
Chứa body code	Có thể	Có thể
Kế thừa	Một lớp chỉ kế thừa được 1 abstract class.	Một lớp chỉ kế thừa được 1 abstract class.
Dùng khi	Quan hệ is a	Khi các lớp có hành vi chung
Contructor	Có	Không
Ứng dụng phổ biến	Làm lớp cha trong hệ thống kế thừa	Mô tả năng lực / hành vi mà lớp có thể thực hiện.

4.OOP

The init block

Là đoạn mã chạy tự động khi đối tượng được tạo ra.

```
class User(val name: String) {
    init { println("User created: $name") }
}
val u = User( name: "Brian")
```

Dùng để kiểm tra dữ liệu, in log, hoặc khởi tạo giá trị ban đầu.

Enum Class

Enum class trong Kotlin là kiểu dữ liệu đặc biệt dùng để định nghĩa một tập hợp các giá trị cố định, giúp code rõ ràng và an toàn hơn so với String hay Int.

Data class

data class là một dạng đặc biệt của class trong Kotlin, được thiết kế để lưu trữ và xử lý dữ liệu một cách ngắn gọn và tự động.

```
data class Person(val name: String, val age: Int)

fun main() {
    // 1 toString() - in thông tin đối tượng
    val p1 = Person( name: "Brian",  age: 20)
    println("toString(): $p1") // Person(name=Brian, age=20)

    // 2 equals() - so sánh nội dung
    val p2 = Person( name: "Brian",  age: 20)
    println("equals(): ${p1 == p2}")   // true

    // 3 hashCode() - mã đại diện cho dữ liệu
    println("hashCode() p1 = ${p1.hashCode()}") // 2420314
    println("hashCode() p2 = ${p2.hashCode()}") // 2420314

    // 4 copy() - sao chép đối tượng, có thể đổi 1 phần dữ liệu
    val p3 = p1.copy(age = 25)
    println("copy(): $p3") // Person(name=Nam, age=25)

    // 5 componentN() - destructuring (tách giá trị)
    val (n, a) = p1
    println("Destructuring → name: $n, age: $a")
    // name: Nam, age: 20
}
```

Kotlin scratch project programmer

DEMO

www.reallygreatsite.com

Kotlin scratch project programmer

THANK YOU

FOR YOUR ATTENTION