Bootcamp - Android

Roshka - Agosto 2022

Getter y setter en Kotlin

Como habíamos comentado los getter y setter en Kotlin se crean cuando se compila nuestras clases Kotlin por lo que estos códigos son equivalentes

```kotlin
class Person {
    var name: String = "defaultValue"
}
```

==

```kotlin
class Person {
    var name: String = "defaultValue"

    // getter
    get() = field

    // setter
    set(value) {
        field = value
    }
}
```

Getter y setter en Kotlin

Si queremos sobreescribir algunos de estos métodos lo hacemos escribiendo los métodos get y set nuevamente

```
class Perro(nombre: String, edad: Int) {
    var nombre = "First property: $nombre".also(::println)
    var edad: Int = 0
    var edadHumana: Int = 0

        set(value) {
            field = value * 7
        }

    fun ladrar(){
        println("Woof!")
    }
}
```

POO: Encapsulación

La encapsulación consiste en hacer que los datos(propiedades) sean modificados únicamente por las funciones destinadas a tal efecto.

La encapsulación permite que los datos conserven un estado válido y consistente

POO: Encapsulación

Ventajas de la encapsulación:

- Ocultar datos
- Reutilización de código
- Métodos get y set
- Mantenimiento de código
- Facilidad a la hora de testear

POO: herencia

La herencia es uno de los conceptos más importantes de la POO

La herencia es la capacidad de una clase para heredar capacidades o propiedades de otra clase en Kotlin

POO: herencia

Al implementar una clase y para reutilizar el código una clase puede extender de otra, heredando el comportamiento de la clase extendida

Ej.: Animal —- > Perro

Persona —> Programador

POO: herencia

```
open class Animal(edad: Int) {
    var edad: Int = 0

    fun serVivo(){
        println("Soy un ser vivo")
    }
}

class Perro(nombre: String, edad: Int): Animal(edad) {
    var nombre = "First property: $nombre".also(::println)
    var edadHumana: Int = 0

        set(value) {
            field = value * 7
        }

    fun ladrar(){
        println("Woof!")
    }
}
```

POO: herencia

La herencia es la característica más esencial de la programación orientada a objetos. Ayuda a reducir la complejidad de escribir códigos muy grandes, gracias a la reutilización de código.