Herencia en Kotlin

Clase derivada sin constructor

```
open class Vehiculo // La clase Vehiculo permite ser extendida
class Coche : Vehiculo() // Coche extiende Vehiculo
```

Clase derivada con constructor primario

```
open class Vehiculo(numeroDeRuedas: Int) // Clase abierta con constructor
primario

/* Si la clase derivada tiene constructor primario debemos llamar al constructor
de la clase base en la declaración */

class Coche(numeroDeRuedas: Int) : Vehiculo(numeroDeRuedas)
```

Sobreescritura

```
open class Forma {

    open fun dibujar() { /*...*/ } // La función dibujar está abierta
    fun fill() { /*...*/ }
}

class Circulo() : Forma() {
    override fun dibujar() { /*...*/ } // En Circulo la sobreescribimos
}
```

Clases abstractas

```
abstract class Poligono { // Una clase abstracta no se puede instanciar

    abstract fun dibujar() // Una función abstracta se debe sobreescribir
}

class Rectangle : Poligono() {
    override fun dibujar() { /*...*/ }
}
```

Interfaces

```
interface ManagerDeUsuario {
    fun cambiarContrasena()
    // Las funciones se pueden implementar por defecto
    fun obtenerNombreDeUsuario() { /*...*/ }
}
```

Modificadores de visibilidad para miembros de una clase

- **private** significa que es visible dentro de esta clase
- **protected** significa que es visible dentro de la clase y de las subclases.
- **internal** significa que cualquiera dentro del módulo de su misma clase puede verlo.
- **public** significa visible por cualquiera.