

Programación orientada a objetos en swift

Clases

```
class forma{
    var lados = Int? //Atributo. Indicamos que el valor es opcional, si no se asigna valor será nil (Equivalente a NULL en este lenguaje)
    init(lados: Int){//Constructor de la clase
        self.lados = lados//El this es Self en Swift
    }
    func descSimple() -> String{//metodo
        return "una forma con \(lados)"
    }
}
var formilla = forma()//instanciamos un objeto de la clase forma
formilla.lados = 7 //Le asignamos un valor al atributo lados
print(formilla.descSimple())
```

En caso de querer limpiar memoria tras borrar una clase podemos utilizar deinit

Herencia

Para indicar herencia simplemente ponemos dos puntos y la clase de la que se hereda. Se pueden sobreescribir funciones añadiendo un "override" antes del func. Se puede llamar al constructor del padre con "super.init(Atributo:Valor)":

```
class Poliedro: forma{

    var nuevoAtributo: Int
    init(lados:Int, nuevoAtributo:Int){
        super.init(lados:lados)//llamamos al constructor del parent
        self.nuevoAtributo = nuevoAtributo
    }
    override func descSimple(){
        print("Hola, soy una función heredada reescrita")
    }
}
```

From:

<http://www.knoppia.net/> - Knoppia

Permanent link:

<http://www.knoppia.net/doku.php?id=swift:programacionorientadaobjetos&rev=1697645394>

Last update: 2023/10/18 16:09

