La Sénia
Institut d'Educació Secundària
Paiporta

Unión Europea
Fondo Social Europeo
El FSE invierte en tu futuro

Programación Orientada a Objetos (POO)

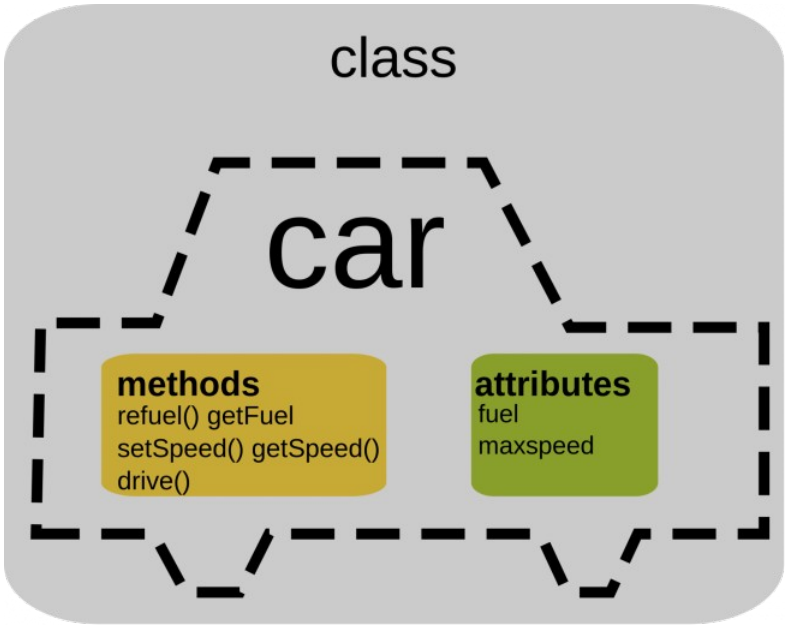
Curso 2023 - 2024

Programación Orientada a Objetos

Modelo de programación con elementos a programar llamados **objetos.**

- Propiedades (**atributos**)
- Acciones (**comportamiento**).

Ejemplo:
Un **coche** es un **objeto**.
Los **Atributos** definen su estado.
Los **métodos** definen cómo se comporta



1. Introducción a la POO

Conceptos:

- Abstracción
- Clases y Objetos
- Atributos y Métodos
- Encapsulamiento
- Constructores
- Getters y Setters

Abstracción

Capacidad de definir las características de los objetos que nos rodean.

Ejemplo. El concepto de **mesa** puede ser distinto para cada persona (mesa de comedor, mesa de clase, mesa de negocios, mesa de cocina o un escritorio)

Abstracción:

"Una mesa se intuye como una superficie plana, sujeta por unas patas y que sirve para realizar tareas como escribir, comer, etc. Puede ser de varios materiales y colores."

2. Abstracción

Propiedades	Acciones
estado	comerEnMesa()
color	prepararMesa()
material	quitarMesa()
numPatas	trabajarEnMesa()
superficie	estudiarEnMesa()

3. Clases y Objetos

Clase

Es una plantilla que define **cómo son los objetos** que pertenecen a ella. Se compone de:

- **Información**: Los **atributos** o propiedades
- **Comportamiento**: Los **métodos** (funciones y/o procedimientos)

Objeto

Es una **instancia de una clase**.

Las propiedades toman valores y los métodos se ejecutan.

Los objetos de la misma clase comparten **propiedades** y **métodos** comunes.

Clase Mesa

Atributos	Métodos
estado	**comerEnMesa()**
color	**prepararMesa()**
material	**quitarMesa()**
numPatas	**trabajarEnMesa()**
superficie	**estudiarEnMesa()**

Objetos:

MesaCocina
Atributos: estado=”libre”, color=”gris”, material=”cristal”, numPatas=4, superficie=”150x70cm”
Métodos: prepararMesa(), quitarMesa() y comerEnMesa())
MesaEstudio
Atributos: estado=”ocupada”, color=”marrón”, material=”madera”, numPatas=2, superficie=”70x40cm”
Métodos: trabajarEnMesa() y estudiarEnMesa())

3. Clases y Objetos

```
class NombreClase {

    //Atributos
    [public | private | protected ] tipoDato nombreVariable;

    //Constructores
    [public | private | protected ] constructor(parametros) {
        //Cuerpo de la función
    }

    //Métodos
    [public | private | protected ] tipoDevuelto nombreMetodoA(parametros) {
        //Cuerpo de la función
    }

    [public | private | protected ] tipoDevuelto nombreMetodoB(parametros) {
        this.nombreVariable=valor;
        this.nombreMetodoA();
    }
}
```

Nota: para hacer referencia a los atributos o métodos de la clase dentro de la clase se usa la palabra **this**

3. Clases y Objetos

```
public class Mesa {

     private String estado;
     private String color;
     private String material;
     prinate int numPatas;
     private String tamañoSuperficie;


    public void prepararMesa() {
          //Código necesario
     }
    public void quitarMesa() {
          //Código necesario
     }
    public void comerEnMesa() {
          //Código necesario
     }
    public void estudiarEnMesa() {
          //Código necesario
     }
    public void trabajarEnMesa() {
          //Código necesario
     }
}
```

Atributos de la clase Mesa

Métodos de la clase Mesa

4. Atributos y Métodos

Atributos

Las características que revelan el estado o cómo es el objeto (**propiedades**)

Métodos

Las acciones que modelan cómo se comporta el objeto e indica lo qué puede hacer (**funciones** y **procedimientos**)

4. Atributos y Métodos

Los **atributos** (**propiedades** o **campos**) almacenan el valor de las propiedades del objeto.

- **Atributos de objeto o instancia**: Cada objeto de la clase toma un valor en el atributo cuando se crea. Este atributo es accesible a través del objeto al que pertenece.
- **Atributos de clase** (se declaran como **static**): Toma el mismo valor para todos los objetos de la clase (es como una variable global para todos los objetso). Es accesible sin necesidad de instanciar la clase.

Atributo de objeto: **MesaCocina.**estado = ”preparada”
Atributo de clase: **Mesa.**numMesas = Mesa.numMesas + 1

4. Atributos y Métodos

Los **métodos** son las **funciones** y **procedimientos** que puede ejecutar el objeto. Podemos distinguir entre:

- **Métodos de objeto o instancia**: Cada objeto invoca a sus métodos con acceso a los atributos de instancia. El método es accesible usando el nombre del objeto o mediante ***this*** desde dentro del propio objeto.
- **Métodos de clase** (se declaran como **static**): No puede acceder a los atributos de instancia ni se puede usar con ***this***. Se accede sin necesidad de instanciar la clase.

Método de objeto: **MesaCocina.**ponerMesa()
Atributo de clase: **Mesa.**devolverTotalMesas()

5. Constructores

- Para crear un objeto o instancia de una clase, se reserva la memoria necesaria para guardar la información indicando los **valores iniciales de sus atributos**.
- Existe un método especial conocido como **Constructor**.

NomClase nombreObjeto = new ConstructorClase(atributos)

- El constructor tiene el mismo nombre que la clase.
- Se puede sobrecargar por si hay atributos que no queremos especificar o desconocemos.

```
public Mesa (String estado, String color, String material, int numPatas,
             String superficie){
    this.estado = estado;
    this.color = color;
    this.material = material;
    this.numPatas = numPatas;
    this.superficie = superficie;
}
```

5. Constructores

Para poder crear los dos objetos:

```
Mesa mesaCocina = new Mesa(“Libre”, Blanca”, “Pino”, 4, ”150x70cm”);

Mesa mesaEstudio = new Mesa (”Marrón”, 2, ”70x40cm”);
```

Sobrecargamos el constructor:

```
public Mesa (){}

public Mesa (String color, int numPatas, String superficie){
   this.color = color;
   this.numPatas = numPatas;
   this.superficie = superficie;
}
```

Encapsulamiento

Los objetos se comportan como una caja negra:

No se debe poder acceder a los atributos de instancia privados si no es a través de los métodos públicos disponibles.

Se pueden aplicar **modificadores** de acceso:

Modificador/Acceso	Clase	Paquete (package)	Subclase	Todos
public	SI	SI	SI	SI
protected	SI	SI	SI	NO
default	SI	SI	NO	NO
private	SI	NO	NO	NO

7. Getters y Setters

Los **atributos** de un objeto deben ser **privados**, no accesibles desde fuera de él.

Para leer o escribir los atributos privados de un objeto se debe recurrir a **métodos públicos** que permitan esas acciones (**Getters** y **Setters**).

Ejemplo: Creamos ***getEstado*** y ***setEstado*** para leer o modificar el atributo ***estado***

```
class Mesa {
    //Atributos
    private String estado;
    //Métodos
    public String getEstado() {
        return this.estado;
    }
    public void setEstado(String nuevoEstado) {
        this.estado = nuevoEstado;
    }
}
```

Ejercicio

Crea una **clase** para los siguientes objetos.

Deberás **abstraer** los **atributos** necesarios así como indicar los **métodos**.

Crea un **constructor** y sobrecárgarlo para poder crear objetos sin atributos o con todos los atributos.

Crea **getters** y **setters** para los atributos privados.