Universidad Europea

UF3: Programación básica

3.3: Introducción a la Programación Orientada a Objetos

Ve más allá

Universidad Europea

Índice

- Clases POJO y Objetos
- Declaración de una clase
- Especificación de una Clase POJO
- Elementos de una clase POJO: Atributos y Métodos
- Elementos de una clase POJO: Atributos y Métodos de Acceso
- Elementos de una clase POJO: Métodos
- Constructores

Universidad Europea

Clases POJO en Java

POJO: Plain Old Java Object

La POO (programación orientada a objetos) da lugar a una colección estructurada de clases.

¿Qué es una clase POJO?

Es una clase que definimos para representar *objetos* (libros, coches...) que nos encontramos en escenarios de la vida real y que necesitamos manejar en programación.

Es un *molde* o plantilla del que luego se pueden crear múltiples objetos.

Contiene **atributos** (variables globales) y **métodos** (funciones/acciones).

Objetos en Java.

¿Qué es un objeto?

Un objeto es una **instancia** de una clase.

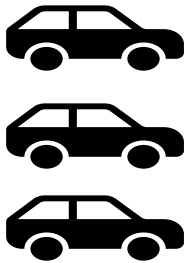
La clase define los **atributos** y **métodos** comunes a los objetos de ese tipo. Luego, cada objeto tendrá sus propios valores en los atributos, y compartirán las mismas funciones.

Debemos crear una clase antes de poder crear objetos (instancias) de esa clase.
Al crear un objeto de una clase, se dice que se crea una instancia de la clase o un objeto propiamente dicho.

Clase

Coche
- matricula: String - velocidad: int
+ acelera(int) + frena (int)

Objetos



matricula: JLS1234
velocidad: 50
matricula: HKN5678
velocidad: 90
matricula: LBC9900
velocidad: 30

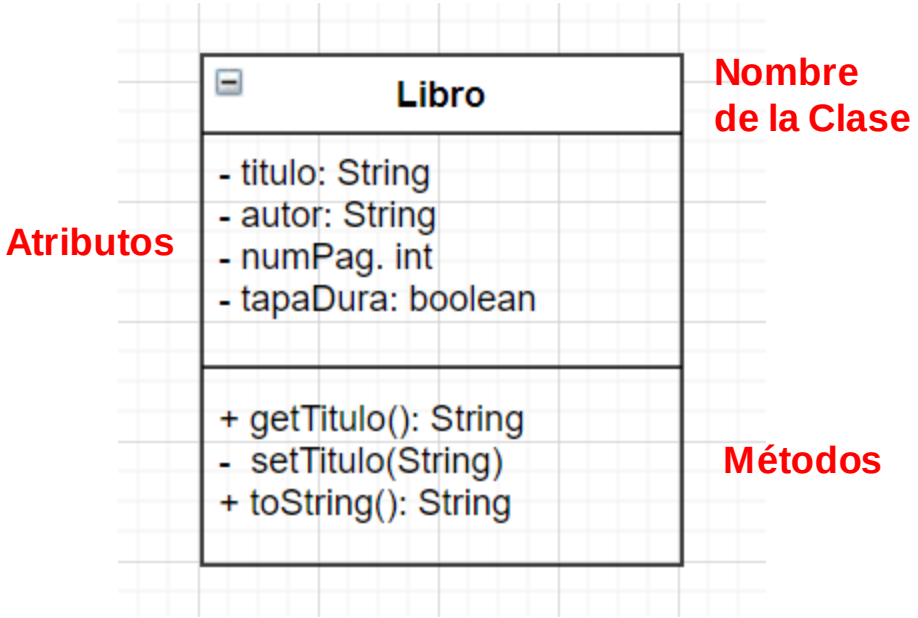
Declaración de una clase

```
[package paquete]
[import otropaquete]

[public] class NombreClase {

   [atributos o variables de la clase]
   [métodos o funciones de la clase]

}
```

Especificación de una clase POJO

Atributos y Métodos

En lenguaje UML, lenguaje utilizado en la realización de un Documento Técnico, se representan las clases del siguiente modo:



Elementos de una clase POJO

Atributos y Métodos

En una clase POJO nos vamos a encontrar atributos y métodos:

- Los **atributos** son **variables globales** para almacenar la información concreta de un objeto.
- Los **métodos** indicarán las **acciones** de un objeto.

Elementos de una clase POJO

Atributos y Métodos

A los elementos de una clase (atributos y métodos) se les puede asignar un **modificador de visibilidad**:

- **`public`**: Elemento público, accesible desde todas las clases
- **`private`**: Elemento privado, accesible sólo dentro de la clase donde se define.

Los métodos pueden tener distintos niveles de visibilidad pero, por principio de diseño, **todos los atributos de una clase son *privados*.**

Elementos de una clase POJO

Atributos y Métodos

Los atributos pueden ser tipos de datos primitivos o tipos de datos de referencia.

Para acceder a los atributos (que, recordemos, *son privados*) utilizamos métodos públicos:

- Métodos **get**, para obtener el valor del atributo.
- Métodos **set**, para establecer o cambiar el valor del atributo.

Elementos de una clase POJO

Métodos

Un método está compuesto por:

- **Cabecera**: Identificador y Parámetros
- **Cuerpo**: Secuencia de instrucciones

Todo método tiene un **valor de retorno** (indicado en la cabecera).
Si no devuelve nada, se indica con **`void`**.

Universidad Europea

Actividad

1. Crear una clase **Libro** con cuatro atributos: uno de tipo String (**titulo**), otro de tipo String (**autor**), otro de tipo entero (**numPag**), y otro de tipo booleano (**leido**). Los atributos deben ser privados.
2. Escribir los métodos de acceso del atributo titulo, **getTitulo** y **setTitulo**.
3. Generar con Eclipse los métodos de acceso para el resto de atributos.
4. Crear una clase ejecutable **UsoLibro** en la que crearemos dos objetos de tipo **Libro**. Asignar valores a los atributos de los dos objetos.
5. Imprimir por consola los valores de los atributos de los dos objetos.

Actividad

- Intentar imprimir por consola directamente el objeto como variable que es. ¿Qué imprime?
- Implementar/sobreescribir el método **toString** heredado de la clase **Object**, en la clase **Libro**. Este método nos permite indicar qué es lo que se debe mostrar con respecto a un objeto cuando invocamos al método `print` o `println` pasándole dicho objeto como parámetro.

```
public String toString() { … }
```

Constructores

Los constructores de una clase son "**métodos**" que **inicializan los valores de los atributos de una clase POJO**.

Si para una clase no se define ningún método constructor se crea uno por defecto.

El **constructor por defecto** es un constructor sin parámetros que no hace nada. Los atributos del objeto son iniciados con los valores predeterminados por el sistema.

Java crea un constructor por defecto *si no hemos definido ninguno en la clase*. Si en una clase definimos un constructor ya no se crea automáticamente el constructor por defecto, por lo tanto si queremos usarlo deberemos crearlo nosotros.

Los constructores tienen el mismo nombre que la clase en la que son definidos y **nunca tienen tipo de retorno, ni especificado, ni void**.

Universidad Europea

Ejemplo de constructor

```
public class Persona
{
    //ATRIBUTOS
    private String nombre;
    private int edad;

    //CONSTRUCTORES
    public Persona() {}    //CONSTRUCTOR SIN PARÁMETROS
    public Persona(String nombre, int edad)
    {
        this.nombre  = nombre;
        this.edad = edad;
    }
}
```

Invocar un constructor

Si creamos un objeto de tipo referencia a la clase Persona invocando el constructor por defecto:

```
Persona persona = new Persona();
```

Los atributos del objeto son iniciados con los valores predeterminados por el sistema, en este caso:

```
nombre = null, edad = 0
```

Invocar un constructor

Si creamos un objeto de tipo referencia a la clase Persona invocando otro constructor:

```
Persona persona = new Persona(“Ana”, 30);
```

Los atributos del objeto son iniciados con los valores que se le pasan como parámetros.

```
nombre = “Ana”, edad = 30
```