Clases y métodos estáticos

Contenido

Introducción a la POO

- ¿Qué es un paradigma de programación?
- ¿Qué es la programación orientada a objetos?
- Pilares de la POO.
- Principio DRY.

Clases

- ¿Qué es una clase?
- Identificadores de las clases.
- Composición de una clase.
- Sintaxis de una clase en Java.

Introducción a la POO

¿Qué es un paradigma de programación?

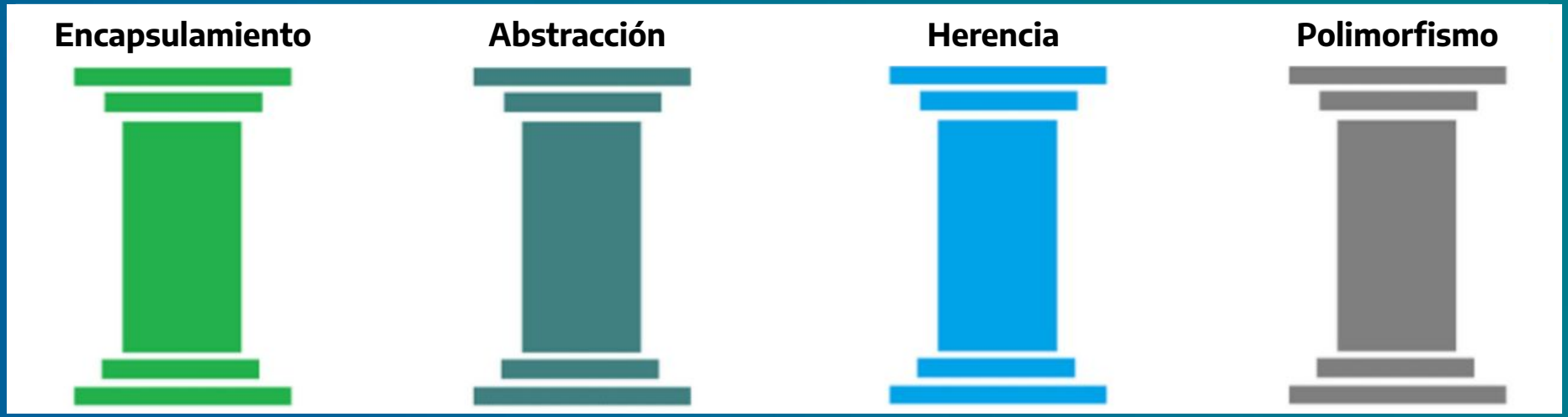
Un **paradigma** es una teoría o conjunto de teorías cuyo núcleo central se acepta sin cuestionar y que suministra la base y modelo para resolver problemas y avanzar en el conocimiento.

Un **paradigma de programación** define la forma, metodología o estilo con el que se resolverá un problema utilizando un lenguaje de programación.

¿Qué es la Programación Orientada a Objetos?

Es un **paradigma de programación** que propone resolver problemas a través de identificar objetos de la vida real, sus **atributos** (datos), su **comportamiento** (acciones) y las **relaciones** de colaboración entre ellos.

Pilares de la POO



Principio DRY

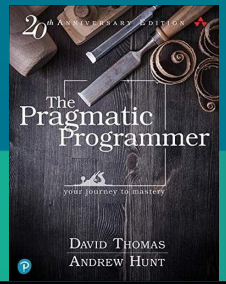
*"Toda **pieza de conocimiento** debe tener una representación **única**, **inequívoca** y **fidedigna** dentro de un sistema."*

The pragmatic programmer - Dave Thomas

Principio DRY

"Many people took it [the DRY principle] to refer to code only: they thought that DRY means 'don't copy-and-paste lines of sources.' [...] DRY is about the duplication of knowledge, of intent. It's about expressing the same thing in two different places, possibly in two totally different ways".
The pragmatic programmer - Dave Thomas

La idea del principio DRY es simple:
Cuando ocurra un cambio no deberíamos necesitar actualizar múltiples cosas en paralelo.



Clases

¿Qué es una CLASE?

Una clase es una **descripción** de un **conjunto de objetos** que comparten los mismos **atributos**, **métodos**, **relaciones** y **semántica** en un determinado **contexto**.

Una clase es una implementación de una abstracción.

Identificadores de clases

A la hora de nombrar clases debemos seguir las siguientes convenciones:

Grafía pascal (UpperCamelCase)
La primera letra del identificador y la primera letra de las siguientes palabras concatenadas están en mayúsculas.

Uso de sustantivos
Los nombres de clases se escriben con sustantivos, ya que representan a objetos.

Nombres descriptivos
Como todo identificador, debe describir con la mayor exactitud posible a lo que representa.

Ejemplos
HistoriaClinica, Mascota, JugadorDeFutbol, LibroDiario, Transferencia, DocenteAdjunto, ExamenFinal.

Composición de una clase

Atributos
Representan **características** que son compartidas por todos los objetos de una clase.
Definen el rango de valores que puede tomar cada una de las propiedades de un objeto.

*Utilizar notación **camelCase** y **sustantivos**.*

Métodos
Un método es **la implementación de una operación**.
Una operación es **una abstracción de algo que puede hacer un objeto** y que es compartido por todos los objetos de esa clase.

*Utilizar notación **camelCase** y **verbos**.*

Composición de una clase

Relaciones

Las clases se conectan entre sí a través de relaciones y **colaboran** para realizar un comportamiento mayor.

Los objetos de clases relacionadas pueden interactuar entre sí.

Semántica

La semántica es el **significado** que se le da a la clase y a sus relaciones dentro del contexto de negocio.

Sintaxis de una clase en Java

```java
public class Persona {

    // ATRIBUTOS
    public String apellido;
    public String nombre;
    public int edad;
    public static String datoEstatico;

    // CONSTRUCTOR
    public Persona(){

        this.apellido = "Doe";
        this.nombre = "John";
        this.edad = 20;
    }

    // MÉTODOS
    public void saludar(){

        System.out.println("Hola, soy " + this.nombre +
                           " " + this.apellido);
        System.out.println("Tengo " + this.edad + " años.");
    }

    public static String metodoEstatico(){

        return Persona.datoEstatico;
    }
}
```

Modificadores de clase

NOMBRE	DESCRIPCIÓN
(*) public	Accesible desde cualquier paquete.
(*) default (sin modificador)	Accesible sólo desde el mismo paquete.
final	La clase no puede ser derivada.
abstract	La clase no puede ser instanciada.

Sintaxis atributos en Java

```
// ATRIBUTOS
public String apellido;
public String nombre;
public int edad;
public static String datoEstatico;
```

Modificadores de atributos

NOMBRE	DESCRIPCIÓN
(*) public	Accesible desde cualquier paquete.
(*) protected	Accesibles desde el mismo paquete y derivadas.
(*) default (sin modificador)	Accesible sólo desde el mismo paquete.
(*) private	Accesible sólo desde la misma clase.
final	Una vez asignada, no podrá modificarse.
static	Pertenece a la clase y no a las instancias

Sintaxis métodos en Java

```
// MÉTODOS
public void saludar(){

    System.out.println("Hola, soy " + this.nombre +
                       " " + this.apellido);
    System.out.println("Tengo " + this.edad + " años.");
}

public static String metodoEstatico(){

    return Persona.datoEstatico;
}
```

Modificadores de métodos

NOMBRE	DESCRIPCIÓN
(*) public	Accesible desde cualquier paquete.
(*) protected	Accesibles desde el mismo paquete y derivadas.
(*) default (sin modificador)	Accesible sólo desde el mismo paquete.
(*) private	Accesible sólo desde la misma clase.
final	No puede ser sobrescrito por clases derivadas.
static	Pertenece a la clase y no a las instancias.
abstract	No tiene implementación y debe ser sobrescrito.

Ejercitación