Polimorfismo con clases abstractas e interfaces

Frida Cano Falcón

Academia Java MTY
Agosto 2024

Viernes 16 de agosto de 2024

Introducción

En el desarrollo de software, el ***polimorfismo*** es un concepto clave que permite a los desarrolladores crear aplicaciones más flexibles y mantenibles. En Java, el polimorfismo se puede implementar a través de clases abstractas e interfaces, proporcionando una manera de definir comportamientos comunes que pueden ser compartidos y personalizados por diferentes tipos de objetos. Este reporte presenta un ejemplo práctico de polimorfismo aplicado en un sistema de administración de cursos en una plataforma educativa. Se utilizarán clases abstractas e interfaces para modelar diferentes tipos de cursos (presenciales y en línea) y las acciones que estos cursos pueden realizar, como evaluar y certificar a los estudiantes.

Esquema del Proyecto

Clases Abstractas:

- ***Curso***: Clase abstracta base que define atributos y métodos comunes para todos los cursos.
- ***CursoPresencial***: Subclase de Curso que representa cursos que se imparten físicamente en un lugar.
- ***CursoEnLinea***: Subclase de Curso que representa cursos que se imparten a través de internet.

***Interfaces*:**

- ***Evaluable***: Interface que define métodos relacionados con la evaluación de los estudiantes.
- ***Certificable***: Interface que define métodos relacionados con la certificación de los estudiantes.

Explicación del Polimorfismo

En este proyecto, se aplica el polimorfismo de varias formas:

- **Polimorfismo a través de Clases Abstractas**: La clase *Curso* es abstracta y define un comportamiento general (*mostrarDetalles*). Las subclases (*CursoPresencial* y *CursoEnLinea*) sobrescriben este método para proporcionar implementaciones específicas.
- **Polimorfismo a través de Interfaces**: Las interfaces *Evaluable* y *Certificable* permiten que diferentes clases implementen métodos específicos (*evaluar* y *certificar*) que son invocados en la clase *Principal* sin importar la implementación exacta.
- **Uso de Casting**: En la clase *Principal*, se utiliza casting para convertir las instancias de *Curso* en *Evaluable* y *Certificable*, permitiendo que se llamen a estos métodos en objetos que implementan estas interfaces.

Implementación de Clases y Interfaces

Interfaz - Evaluable

src > com > curso > tarea > Evaluable.java > ...

```java
package com.curso.tarea;

public interface Evaluable {
    void evaluar();
}

```

Interfaz - Certificable

src > com > curso > tarea > Certificable.java > Certificable

```java
package com.curso.tarea;

public interface Certificable {
    void certificar();
}

```

Clase abstracta - Curso

src > com > curso > tarea > Curso.java > Curso > mostrarDetalles()

```java
package com.curso.tarea;

public abstract class Curso {
    protected String nombre;
    protected String instructor;
    protected int duracion; // duración en horas

    public Curso(String nombre, String instructor, int duracion) {
        this.nombre = nombre;
        this.instructor = instructor;
        this.duracion = duracion;
    }

    public abstract void mostrarDetalles();
}

```

Subclase - Curso En Línea

src > com > curso > tarea > CursoPresencial.java > CursoPresencial

```java
package com.curso.tarea;

public class CursoPresencial extends Curso implements Evaluable {

    private String ubicacion;

    public CursoPresencial(String nombre, String instructor, int duracion, String ubicacion) {
        super(nombre, instructor, duracion);
        this.ubicacion = ubicacion;
    }

    @Override
    public void mostrarDetalles() {
        System.out.println("Curso Presencial: " + nombre);
        System.out.println("Instructor: " + instructor);
        System.out.println("Duración: " + duracion + " horas");
        System.out.println("Ubicación: " + ubicacion);
    }

    @Override
    public void evaluar() {
        System.out.println("Evaluando a los estudiantes del curso presencial: " + nombre);
    }
}
```

Subclase - Curso Presencial

src > com > curso > tarea > CursoEnLinea.java > CursoEnLinea

```java
public class CursoEnLinea extends Curso implements Evaluable, Certificable {
    public CursoEnLinea(String nombre, String instructor, int duracion, String plataforma) {
        super(nombre, instructor, duracion);
        this.plataforma = plataforma;
    }

    @Override
    public void mostrarDetalles() {
        System.out.println("Curso en Línea: " + nombre);
        System.out.println("Instructor: " + instructor);
        System.out.println("Duración: " + duracion + " horas");
        System.out.println("Plataforma: " + plataforma);
    }

    @Override
    public void evaluar() {
        System.out.println("Evaluando a los estudiantes del curso en línea: " + nombre);
    }

    @Override
    public void certificar() {
        System.out.println("Certificando a los estudiantes del curso en línea: " + nombre);
    }
}
```

Clase Principal

src > com > curso > tarea > Principal.java > Principal > main(String[])

```java
package com.curso.tarea;

public class Principal {

    Run | Debug
    public static void main(String[] args) {
        Curso presencial = new CursoPresencial(nombre:"Programación en Java", instructor:"Juan Pérez", duracion:40, u
        Curso enLinea = new CursoEnLinea(nombre:"Introducción a la IA", instructor:"María Gómez", duracion:30, plataf

        presencial.mostrarDetalles();
        ((Evaluable) presencial).evaluar();

        System.out.println();

        enLinea.mostrarDetalles();
        ((Evaluable) enLinea).evaluar();
        ((Certificable) enLinea).certificar();
    }
}

```

Resultados de las pruebas

```
PS C:\Users\HP\Documents\GitHub\EntregablesJavaAcademy2024\Semana 1\Polimorfismo>  c:; cd 'c:\Users\HP\Documents\GitHub\EntregablesJavaAcade
my2024\Semana 1\Polimorfismo'; & 'C:\Program Files\Java\jdk-17\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\Do
cuments\GitHub\EntregablesJavaAcademy2024\Semana 1\Polimorfismo\bin' 'com.curso.tarea.Principal'
Curso Presencial: Programación en Java
Instructor: Juan Pérez
Duración: 40 horas
Ubicación: Aula 101
Evaluando a los estudiantes del curso presencial: Programación en Java

Curso en Línea: Introducción a la IA
Instructor: María Gómez
Duración: 30 horas
Plataforma: Zoom
Evaluando a los estudiantes del curso en línea: Introducción a la IA
Certificando a los estudiantes del curso en línea: Introducción a la IA
```