DONCAN ALEJANDRO CASQUETE OLIVEROS

PATRONES DE DISEÑO DE SOFTWARE

Patrón Singleton

Patrón Singleton en Java

# ¿Qué es el patrón Singleton?

El patrón Singleton es un patrón de diseño creacional que garantiza que una clase tenga una única instancia y proporciona un punto de acceso global a esa instancia. Es útil cuando se necesita un único objeto que coordine las acciones a través del sistema, como una conexión a base de datos, un logger o una clase de configuración global.

## Características del Singleton

- Constructor privado para evitar múltiples instancias.

- Una variable estática que almacena la única instancia de la clase.

- Un método público y estático que devuelve la instancia única.

# Ejemplo: Clase Conexion.java

Código de la clase:

package App;  
  
public class Conexion {  
 private static Conexion instancia;  
  
 private Conexion() {  
 }  
  
 public static Conexion getInstancia() {  
 if (instancia == null) {  
 instancia = new Conexion();  
 }  
 return instancia;  
 }  
  
 public void conectar() {  
 System.out.println("Me conecte a la BD");  
 }  
  
 public void desconectar() {  
 System.out.println("Me desconecte de la BD");  
 }  
}

## Explicación de Conexion.java

1. La clase tiene una variable estática 'instancia' que guardará la única instancia de la clase.  
2. El constructor es privado, por lo tanto, no se puede crear una instancia desde fuera de la clase.  
3. El método 'getInstancia()' verifica si la instancia ya existe. Si no, la crea. Luego devuelve esa instancia.  
4. Los métodos 'conectar()' y 'desconectar()' son ejemplos para mostrar el uso del Singleton.

# Ejemplo: Clase App.java

Código de la clase:

package Inpahu;  
  
import App.Conexion;  
  
public class App {  
  
 public static void main(String[] args) {  
 Conexion c = Conexion.getInstancia();  
 c.conectar();  
 c.desconectar();  
  
 boolean respuesta = c instanceof Conexion;  
 System.out.println(respuesta);  
 }  
}

## Explicación de App.java

1. En lugar de crear un objeto con 'new', se llama a 'Conexion.getInstancia()', lo cual garantiza una única instancia.  
2. Se llaman los métodos 'conectar()' y 'desconectar()' para probar la funcionalidad de la instancia.  
3. Se imprime el resultado de verificar si 'c' es una instancia de la clase 'Conexion', lo cual debe retornar 'true'.