Ejercicios Patrones

1. Command

Explicación breve:

El patrón Command encapsula una solicitud como un objeto, permitiendo parametrizar clientes con diferentes solicitudes, encolar o registrar solicitudes, y soportar operaciones reversibles.

Enunciado:

Simula un mando con tres botones:

- Encender luz
- Apagar luz
- Activar ventilador

Implementa:

- Interfaz Command
- Comandos: LightOnCommand, LightOffCommand, FanOnCommand
- Clases receptoras: Light, Fan
- Clase RemoteControl que almacene y ejecute comandos

2. DTO (Data Transfer Object)

Explicación breve:

Un DTO es un objeto simple que transfiere datos entre capas, sin lógica de negocio.

Enunciado:

Crea una clase Customer con nombre, correo y teléfono. Luego crea una clase CustomerDTO y un método que convierta un Customer en un CustomerDTO.

3. Facade

Explicación breve:

El patrón Facade proporciona una interfaz simplificada a un subsistema complejo.

Enunciado:

Diseña una fachada para un sistema de cine que incluye:

- Proyector
- Pantalla
- Sistema de sonido

La clase CinemaFacade debe tener un método watchMovie() que lo prepare todo.

4. Factory Method

Explicación breve:

Este patrón define una interfaz para crear objetos, pero permite que las subclases decidan qué clase instanciar.

Enunciado:

Crea una fábrica para vehículos:

- Car y Motorcycle implementan Vehicle
- VehicleFactory tiene un método createVehicle(String type)

5. Singleton

Explicación breve:

Este patrón garantiza que una clase tenga solo una instancia y proporcione un punto de acceso global a ella.

Enunciado:

Crea una clase ConfigManager que solo permita una instancia y contenga parámetros de configuración.

6. Strategy

Explicación breve:

Permite definir una familia de algoritmos, encapsular cada uno y hacerlos intercambiables sin modificar el código cliente.

Enunciado:

Crea una clase PaymentContext que pueda utilizar diferentes estrategias de pago (PayPal, CreditCard).

Soluciones
Agenda 2025