PATRON DE DISENO BUILDER

CONCEPTOS Y USOS

AARON HERNANDEZ GARCIA
PATRONES DE DISENO

¿QUE ES EL PATRON BUILDER?

EL PATRON BUILDER ES UN PATRóN DE DISENO CREACIONAL QUE SE UTILIZA PARA CONSTRUIR OBJETOS COMPLEJOS PASO A PASO.

PROPOSITO: SU OBJETIVO PRINCIPAL ES SEPARAR LA CONSTRUCCION DE UN OBJETO DE SU REPRESENTACIóN, PERMITIENDO CREAR DIFERENTES REPRESENTACIONES UTILIZANDO EL MISMO PROCESO DE CONSTRUCCION.

PROBLEMATICA

UNA AGENCIA DE VIAJES DESEA OFRECER PAQUETES PERSONALIZADOS A SUS CLIENTES. CADA CLIENTE TIENE NECESIDADES ESPECIFICAS: ALGUNOS DESEAN INCLUIR TRANSPORTE, ALOJAMIENTO Y ACTIVIDADES; OTROS SOLO TRANSPORTE Y ACTIVIDADES, O SOLO ALOJAMIENTO. EL SISTEMA ACTUAL TIENE CODIGO REDUNDANTE PARA MANEJAR MULTIPLES CONFIGURACIONES DE PAQUETES, LO QUE DIFICULTA SU MANTENIMIENTO Y AMPLIACION.

COMO RESOLVERLO CON BUILDER:

EL PATRON BUILDER PUEDE UTILIZARSE PARA ESTRUCTURAR EL PROCESO DE CREACION DE PAQUETES DE VIAJE.

SOLUCION

- DIVIDIR EL PROCESO EN PASOS:
  - SELECCIONAR EL TRANSPORTE.
  - ELEGIR EL ALOJAMIENTO.
  - ANADIR ACTIVIDADES OPCIONALES.
  - CALCULAR EL COSTO TOTAL.

- ELEMENTOS DEL PATRON BUILDER:
  - PRODUCTO (PAQUETE DE VIAJE): EL OBJETO FINAL QUE CONTIENE TRANSPORTE, ALOJAMIENTO Y ACTIVIDADES.
  - BUILDER (INTERFACE): DEFINE LOS METODOS PARA CONSTRUIR CADA PARTE DEL PAQUETE.
  - CONCRETE BUILDERS: IMPLEMENTACIONES ESPECÍFICAS, POR EJEMPLO:
    - BUILDER PARA UN PAQUETE ECONOMICO.
    - BUILDER PARA UN PAQUETE PREMIUM.
  - DIRECTOR: ORQUESTA EL PROCESO DE CONSTRUCCION BASANDOSE
  - EN LAS OPCIONES SELECCIONADAS POR EL CLIENTE.

CODIGO

travel.cs > TravelAgent > BuildEconomyPackage

```
// Producto final: Paquete de Viaje
6 references
public class TravelPackage
{
    3 references
    public string Transport { get; set; }
    3 references
    public string Accommodation { get; set; }
    3 references
    public List<string> Activities { get; set; } = new();

    0 references
    public override string ToString()
    {
        return $"Transport: {Transport}, Accommodation: {Accommodation}, Activities: {string.Join(", ", Activities)}";
    }
}

```

CODIGO

```
// Interfaz Builder
4 references
public interface ITravelPackageBuilder
{
    4 references
    void SetTransport(string transport);
    4 references
    void SetAccommodation(string accommodation);
    5 references
    void AddActivity(string activity);
    3 references
    TravelPackage GetPackage();
}

// Concrete Builder: Paquete Económico
1 reference
public class EconomyTravelPackageBuilder : ITravelPackageBuilder
{
    4 references
    private TravelPackage _package = new();

    3 references
    public void SetTransport(string transport) => _package.Transport = transport;
    3 references
    public void SetAccommodation(string accommodation) => _package.Accommodation = accommodation;
    4 references
    public void AddActivity(string activity) => _package.Activities.Add(activity);
    2 references
    public TravelPackage GetPackage() => _package;
}
```

CODIGO

```
// Concrete Builder: Paquete Premium
1 reference
public class PremiumTravelPackageBuilder : ITravelPackageBuilder
{
    4 references
    private TravelPackage _package = new();

    3 references
    public void SetTransport(string transport) => _package.Transport = transport;
    3 references
    public void SetAccommodation(string accommodation) => _package.Accommodation = accommodation;
    4 references
    public void AddActivity(string activity) => _package.Activities.Add(activity);
    2 references
    public TravelPackage GetPackage() => _package;
}
```

TERMINAL

```
Economy Package:
Transport: Bus, Accommodation: Standard Hotel, Activities: City Tour

Premium Package:
Transport: Airplane, Accommodation: Luxury Resort, Activities: Snorkeling, Private Island Tour

[Done] exited with code=0 in 9.965 seconds
```