Práctica 04

Modelado y Programación 2022-2

"Paw patrols"

Castellanos Palacios Gerardo 318245630
Vargas Gutiérrez Julieta 318341945

Abril 2022

1. Teoría

1.1. Factory

El patrón de diseño factory hace la creación de objetos sin exponer la lógica detrás, donde el cliente utiliza una interfaz en común para crear un nuevo tipo de objeto, permitiendo que las subclases alteren el tipo de objetos que se harán. Se utiliza una función estática que crea y devuelve la instancia.

- Desventajas

    1. Requiere un número elevado de clases y la extensión del código será elaborada. Puede resultar complicado leer el código ya que al delegar funciones puede ser complejo encontrar la mecánica del funcionamiento.

1.2. Abstract Factory

Abstract Factory busca agrupar un conjunto de clases que tiene un funcionamiento en común, sin especificar explícitamente su comportamiento, estas serán las familias que funcionan entorno a una superfabrica. La implementación del patrón nos proporciona un marco que nos permite crear objetos que siguen un patrón general.

- Desventajas

    1. Al tener muchas interfaces y clases, puede introducir complejidad en el código, además es aplicable solo a familias de objetos relacionados. Reduce la legibilidad del código debido al nivel de abstracción que tiene.

1.3. Builder

Builder construye un objeto complejo usando objetos simples y utilizando un enfoque paso a paso, la construcción es controlada por un objeto director que sólo necesita saber el tipo de objeto que va a crear.

- Desventajas

    1. Al tener que crear un builder para cada representación de un producto, se puede acabar con demasiadas clases y la complejidad general del código aumenta.

2. Práctica

Descripción de la práctica

Con los temas vistos *Factory*, *Abstrac Factory* y *Builder* tuvimos que escoger unos de estos patrones para implementar un sistema que pueda crear naves espaciales las cuales tienen: sistema de propulsión, blindaje, cabina y armas. Cada una de estas tienen nombre, descripción y precio, además estos componentes tienen cualidades especificas. Elegimos usar *Abstract Factory* ya que al tener familias de fabricas nos pareció que tenía la estructura más conveniente para solucionar el problema, así podríamos crear cada fabrica para cada componente de la nave.

Incovenientes

Nos resulto complicado diferenciar las características de cada patrón, así como cual sería el conveniente para está práctica.

Compilación y Ejecución

Descomprimir la carpeta zip, colocarse en la carpeta src y en la consola ejecutar

javac Main.java

En la siguiente linea

java Main.java