Patrones de diseño Creacionales en JAVA

Singleton

- Definición
  - El patrón Singleton garantiza que una clase tenga una única instancia y proporciona un punto de acceso global a dicha instancia. Esto es útil en situaciones donde se necesita controlar el acceso a algún recurso compartido.
- Uso
  - Control de acceso a recursos
    - Asegura que solo haya una instancia del recurso, evitando la creación de múltiples instancias que podrían causar inconsistencias.
  - Ejemplo: Logger
    - Un Logger que se encarga de registrar información en archivos o en la salida estándar puede implementarse como un Singleton, garantizando que todos los componentes de un sistema utilicen la misma instancia para registrar eventos.

Factory Method

- Definición
  - El patrón Factory Method define una interfaz para crear objetos, pero permite que las subclases decidan qué clase instanciar. Este enfoque delega la responsabilidad de la creación de objetos a las subclases.
- Uso
  - Delegar la creación de objetos a subclases
    - Permite que cada subclase implemente la lógica de creación, facilitando la extensión del programa sin modificar el código existente.
  - Ejemplo: Creación de botones en GUI
    - En una aplicación de interfaz gráfica, un Factory Method puede ser utilizado para crear diferentes tipos de botones (por ejemplo, botones de Windows, botones de Mac) dependiendo del sistema operativo.

Abstract Factory

- Definición
  - El patrón Abstract Factory proporciona una interfaz para crear familias de objetos relacionados o dependientes sin especificar sus clases concretas. Esto permite crear sistemas con diferentes configuraciones de productos.
- Uso
  - Crear sistemas complejos sin especificar clases concretas
    - Facilita la creación de productos que están diseñados para funcionar juntos, lo que puede ser crucial en aplicaciones que requieren una alta cohesión.
  - Ejemplo: Diferentes temas de GUI
    - Una aplicación puede tener diferentes temas visuales (como claro y oscuro), y el Abstract Factory puede proporcionar las instancias adecuadas de los componentes gráficos para cada tema.

Builder

- Definición
  - El patrón Builder separa la construcción de un objeto complejo de su representación, lo que permite crear diferentes representaciones del mismo tipo de objeto utilizando el mismo proceso de construcción.
- Uso
  - Crear objetos complejos paso a paso
    - Permite construir objetos complejos de una manera más controlada al descomponer el proceso de construcción en pasos específicos.
  - Ejemplo: Construcción de una casa
    - Un Builder puede ser utilizado para crear una casa, donde se pueden definir los pasos para añadir habitaciones, puertas, ventanas, etc., y al final obtener la representación completa de la casa.