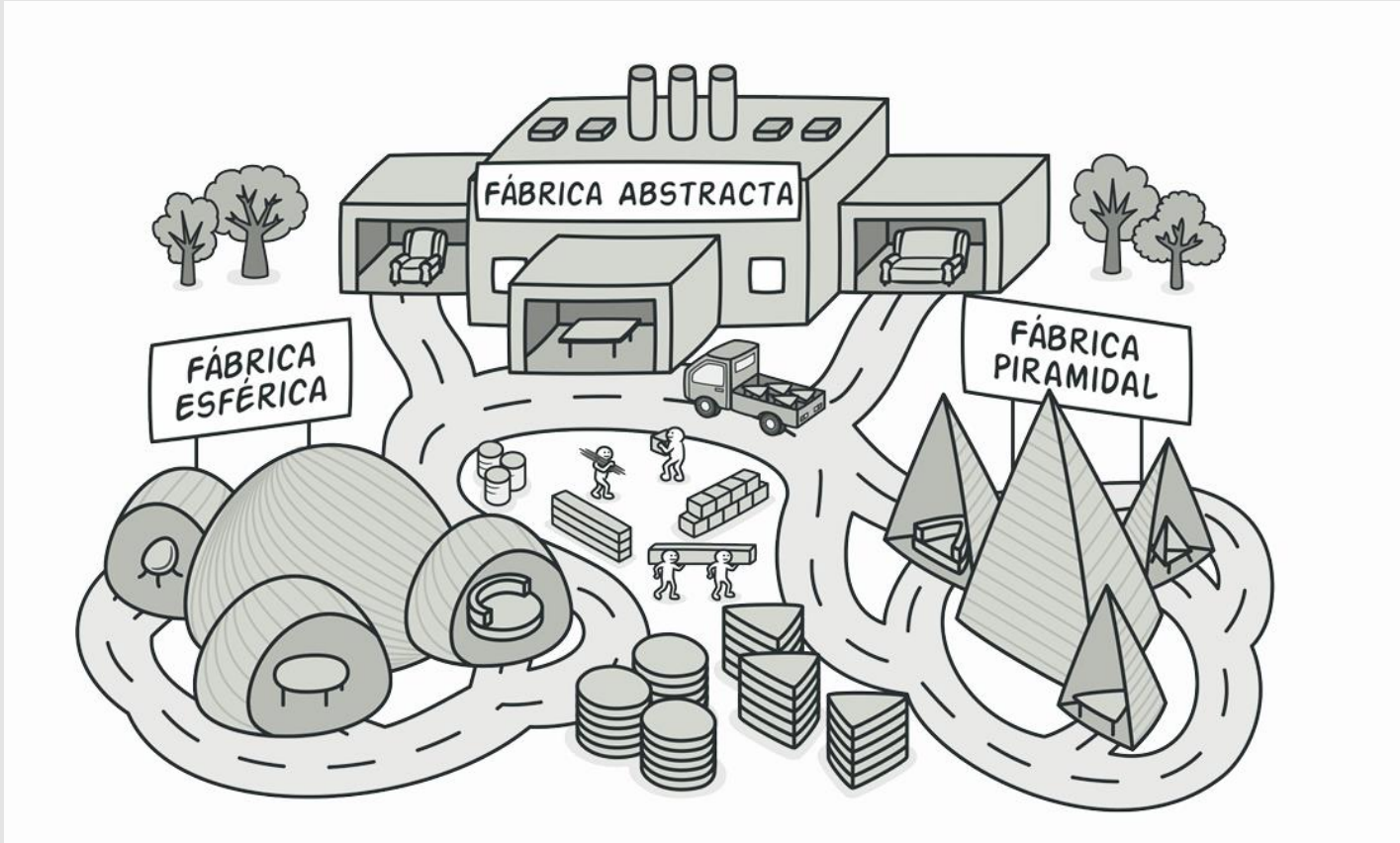
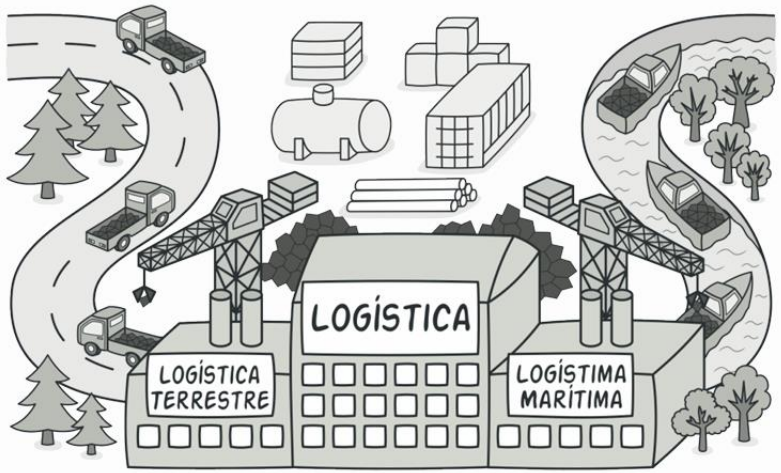
Patrón
Factory Method
Abstract Factory



Shvets, A. (2018). Sumérgete en los patrones de diseño.

¿Qué es el Factory Method?

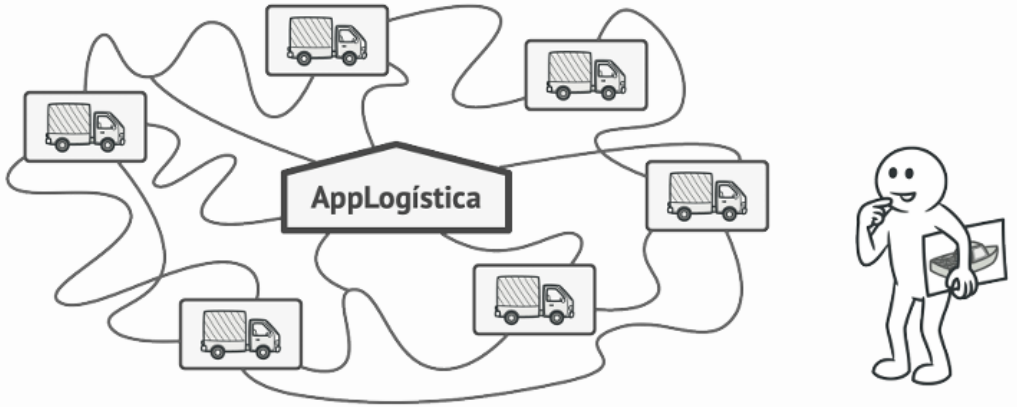
El Factory Method es un patrón de diseño de software que ofrece una interfaz para crear un objeto, pero deja que las subclases decidan qué clase instanciar. Facilita la inclusión de nuevos tipos de productos sin alterar el código que ya utiliza el creador.



Shvets, A. (2018). Sumérgete en los patrones de diseño.

¿Cómo funciona?

Se define una interfaz o clase abstracta con un método para crear objetos, conocido como el "método de fábrica". Las subclases implementan este método para crear objetos de clases concretas. De esta manera, el proceso de creación se encapsula en las subclases, permitiendo flexibilidad en la adición de nuevos tipos de objetos.

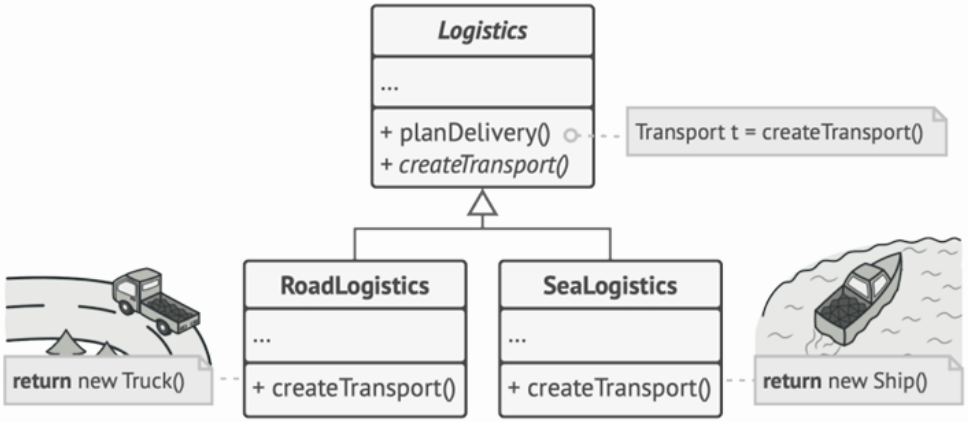


Añadir una nueva clase al programa no es tan sencillo si el resto del código ya está acoplado a clases existentes.

Shvets, A. (2018). Sumérgete en los patrones de diseño.

¿Para qué se utiliza?

Es ideal para situaciones donde hay una estructura o producto base, pero se requiere la flexibilidad para extender o modificar los tipos de productos que se crean. Se utiliza en sistemas donde se espera que evolucionen con el tiempo, añadiendo nuevas clases de objetos sin modificar el código cliente.



Las subclases pueden alterar la clase de los objetos devueltos por el método fábrica.

Shvets, A. (2018). Sumérgete en los patrones de diseño.

Tener en cuenta

El Factory Method es un patrón de diseño que promueve la encapsulación y la flexibilidad en la creación de objetos. Permite que las subclases decidan qué objetos concretos crear, facilitando la extensión del sistema con nuevos tipos de objetos sin alterar el funcionamiento existente. Es una herramienta clave en el desarrollo orientado a objetos para gestionar la creación de objetos de manera escalable.

¿Qué es la Abstract Factory?

La Abstract Factory es un patrón de diseño de software que proporciona una interfaz para crear familias de objetos relacionados sin especificar sus clases concretas.

Familias de productos y sus variantes.

Shvets, A. (2018). Sumérgete en los patrones de diseño.

¿Cómo funciona?

Define interfaces para cada tipo de objeto a crear, con clases concretas que implementan esas interfaces para cada familia de objetos. Los clientes pueden solicitar objetos de una familia específica a través de la interfaz de la Abstract Factory, sin conocer los detalles de su implementación.

¿Para qué se utiliza?

Es útil cuando se necesita garantizar que los objetos creados sean compatibles entre sí o cuando se desea ocultar la lógica de creación de objetos detrás de una interfaz común. Se emplea en sistemas donde la configuración o la elección de clases concretas pueden variar según las condiciones del entorno o las preferencias del usuario.

Un sofá de estilo moderno no combina con unas sillas de estilo victoriano.

Shvets, A. (2018). Sumérgete en los patrones de diseño.

Tener en cuenta

La Abstract Factory simplifica la creación de objetos complejos al proporcionar métodos para crear familias completas de objetos relacionados, abstrayendo a los clientes de los detalles de implementación. Es una herramienta poderosa en el desarrollo de software orientado a objetos para gestionar la creación de objetos de manera flexible y eficiente.

Comparación

Característica	Abstract Factory	Factory Method
Propósito	Crear familias de objetos relacionados sin especificar sus clases concretas.	Crear objetos, dejando que las subclases decidan qué clase instanciar.
Flexibilidad	Proporciona alta flexibilidad al permitir la creación de varias familias de objetos.	Ofrece flexibilidad en la creación de objetos, pero se centra en un solo producto a la vez.
Nivel de abstracción	Alto, maneja familias completas de productos.	Medio, se centra en la creación de un tipo de objeto a la vez.
Implementación	Requiere una mayor cantidad de clases e interfaces para implementar las familias de objetos.	Requiere menos clases e interfaces que la Abstract Factory, ya que se enfoca en un producto individual.
Uso	Utilizado cuando hay varios objetos relacionados que deben ser creados juntos.	Utilizado cuando un objeto puede tener múltiples variantes, pero solo una se necesita a la vez.
Personalización	Los clientes trabajan con productos a través de interfaces comunes sin preocuparse por las implementaciones concretas.	Las subclases deciden cómo implementar y personalizar el objeto concreto que se está creando.
Ejemplo de uso	Un sistema de UI que puede cambiar entre diferentes temas de widgets, donde cada tema es una familia de objetos (botones, ventanas, etc.).	Un lector de documentos donde cada tipo de documento (PDF, DOCX) requiere una clase diferente para su creación, pero todos implementan una interfaz común.

Visitar

https://github.com/code-corhuila/electiva-ii-2024-a/blob/main/07-Sesion-19-03/Ejemplo.md

Actividad 1

Service
ABaseService: Explore la creación del archivo BaseService, para ello aplique el concepto visto. Recomendación usar objeto T, D y S
EstudianteService: Use el servicio abstracto

Controller
ABaseController: Explore la creación del archivo BaseController, para ello aplique el concepto visto. Recomendación usar objeto T, D y S
EstudianteController: Use el controller abstracto

Descripción de la actividad

Primero, desarrolle el código fuente y, a continuación, elabore un video tutorial explicando cómo funciona esta arquitectura en cada una de sus capas.

Incluya una sección en la que reflexione sobre la complejidad, así como las ventajas y desventajas de adoptar este enfoque en comparación con métodos tradicionales.

Ejemplo del .md

Referencias

Shvets, A. (2018). Sumérgete en los patrones de diseño.

Docente
Jesús Ariel González Bonilla