Java Design Patterns

Factory Method

Тема

Factory Method

Суть паттерна

Фабричный метод

Фабричный метод — это порождающий паттерн проектирования, который определяет общий интерфейс для создания объектов в суперклассе, позволяя подклассам изменять тип создаваемых объектов.

Проблема

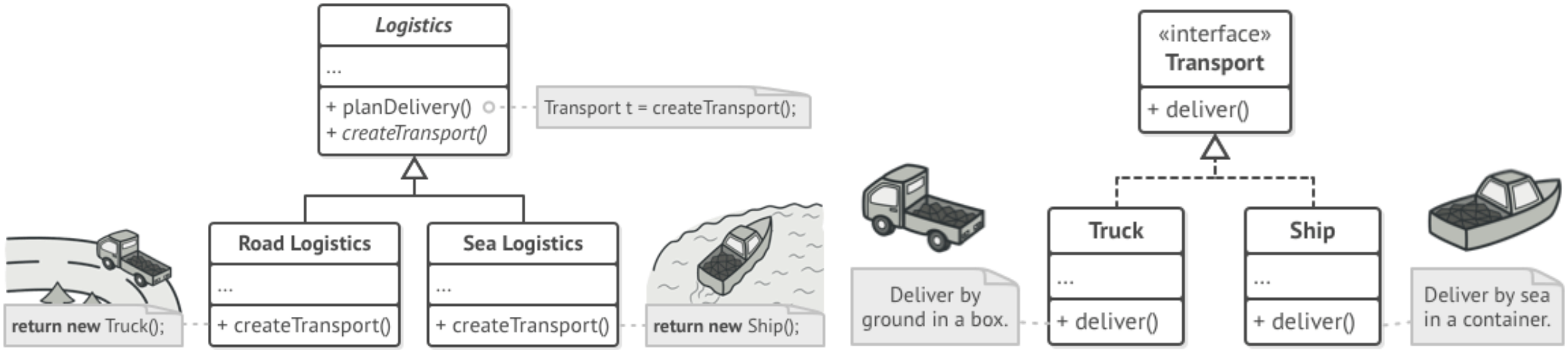
Постановка проблемы

Представьте, что вы создаёте программу управления грузовыми перевозками.

Решение проблемы

Реализация решения

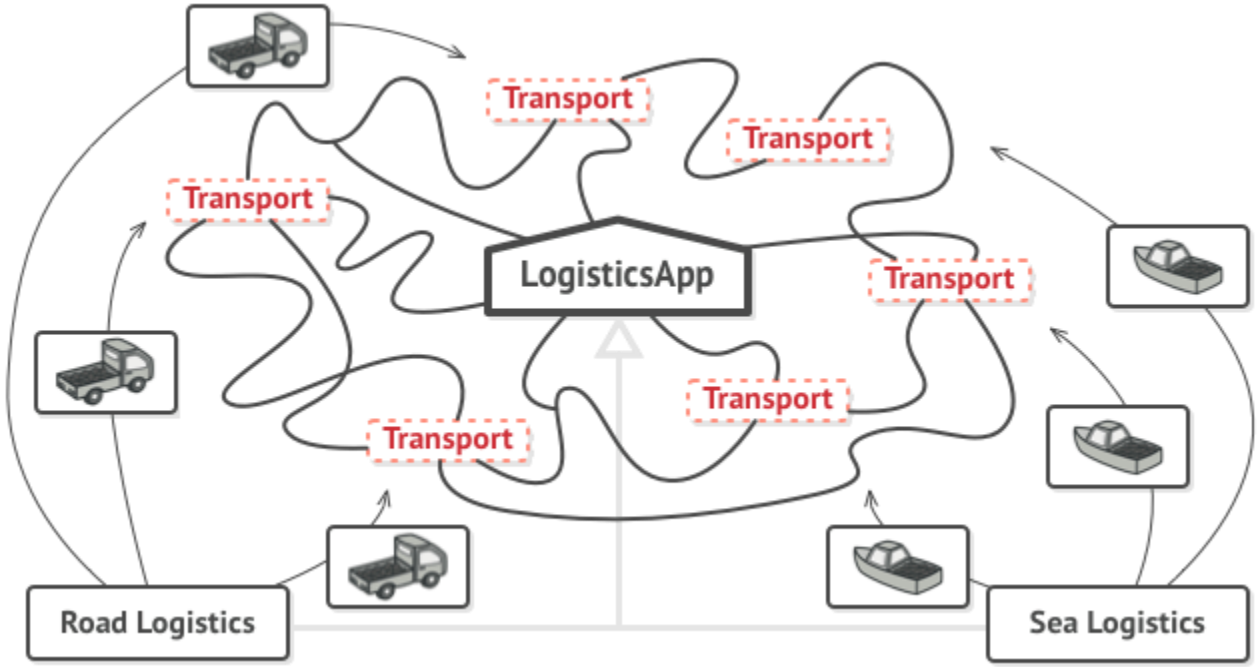
Паттерн Фабричный метод предлагает создавать объекты не напрямую, используя оператор new, а через вызов особого фабричного метода. Не пугайтесь, объекты всё равно будут создаваться при помощи new, но делать это будет фабричный метод.



Решение проблемы

Реализация решения

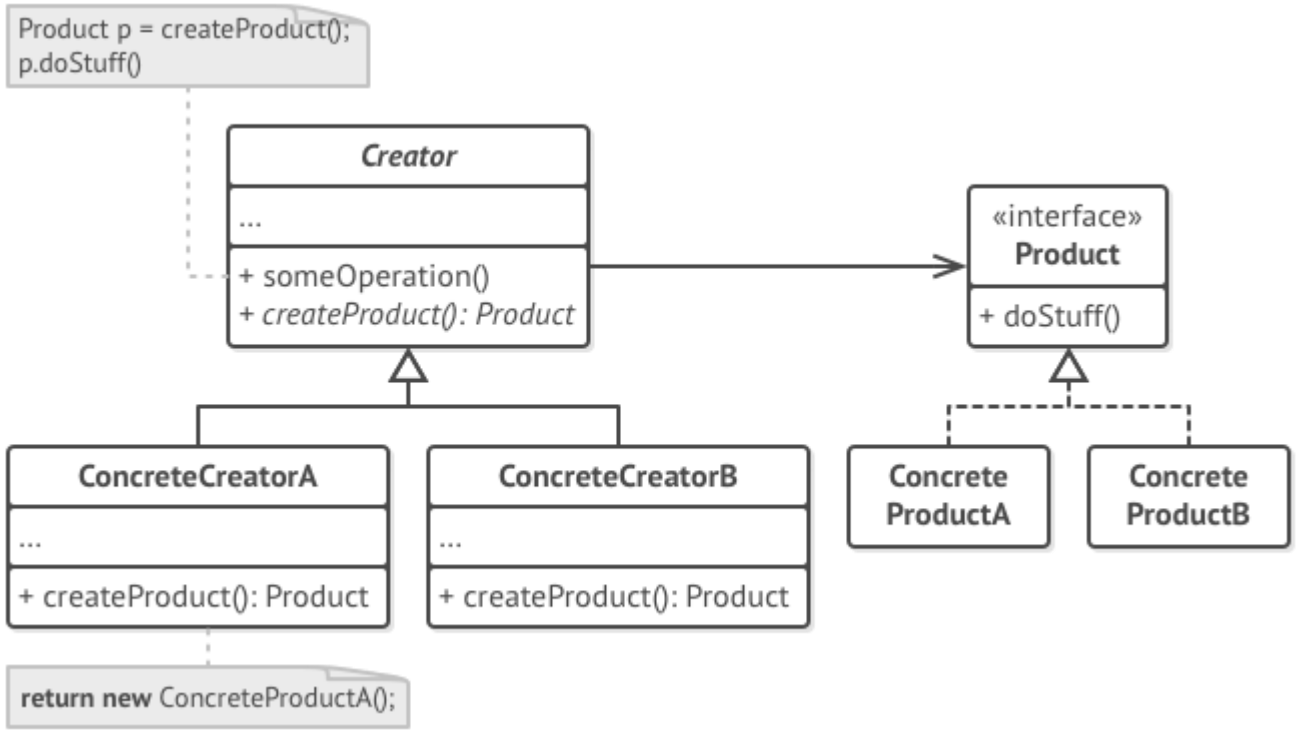
Классы Грузовик и Судно реализуют интерфейс Транспорт с методом доставить. Каждый из этих классов реализует метод по-своему: грузовики везут грузы по земле, а судна — по морю. Фабричный метод в классе Дорожной Логистики вернёт грузовик, а класс Морской Логистики — судно.



Структура

Структура фабричного метода

1. Продукт определяет общий интерфейс объектов, которые может произвести создатель и его подклассы.
2. Конкретные продукты содержат код различных продуктов. Продукты будут отличаться реализацией, но интерфейс у них будет общий.
3. Создатель объявляет фабричный метод, создающий объекты через общий интерфейс продуктов.
4. Конкретные создатели по-своему реализуют фабричный метод, производя те или иные конкретные продукты.



Применимость

Использование паттерна

1. Когда заранее неизвестны типы и зависимости объектов, с которыми должен работать ваш код
2. Когда вы хотите дать возможность пользователям расширять части вашего фреймворка или библиотеки
3. Когда вы хотите экономить системные ресурсы, повторно используя уже созданные объекты, вместо создания новых

Шаги реализации

Алгоритм реализации паттерна

1. Приведите все создаваемые продукты к общему интерфейсу.
2. В классе, который производит продукты, создайте пустой фабричный метод. В качестве возвращаемого типа укажите общий интерфейс продукта.
3. Затем, пройдитесь по коду класса и найдите все участки, создающие продукты. Поочерёдно замените эти участки вызовами фабричного метода, перенося в него код создания различных продуктов.
В фабричный метод, возможно, придётся добавить несколько параметров, контролирующих какой из продуктов нужно создать.
4. Для каждого типа продуктов заведите подкласс и переопределите в нём фабричный метод. Переместите туда код создания соответствующего продукта из суперкласса.
5. Если создаваемых продуктов слишком много для существующих подклассов создателя, вы можете подумать о введении параметров в фабричный метод, которые позволят возвращать различные продукты в пределах одного подкласса.
6. Если после всех перемещений фабричный метод стал пустым, можете сделать его абстрактным. Если в нём что-то осталось — не беда, это будет его реализацией по умолчанию

Преимущества и недостатки

Плюсы и недостатки паттерна

Плюсы:

- Избавляет класс от привязки к конкретным классам продуктов.
- Выделяет код производства продуктов в одно место, упрощая поддержку кода.
- Упрощает добавление новых продуктов в программу.
- Реализует *принцип открытости/закрытости*.

Минусы:

- Может привести к созданию больших параллельных иерархий классов, так как для каждого класса продукта надо создать свой подкласс создателя.

Отношения с другими паттернами

Комбинации паттернов проектирования

- Многие архитектуры начинаются с применения Фабричного метода (более простого и расширяемого через подклассы) и эволюционируют в сторону Абстрактной фабрики, Прототипа или Строителя (более гибких, но и более сложных).
- Классы Абстрактной фабрики чаще всего реализуются с помощью Фабричного метода, хотя они могут быть построены и на основе Прототипа.
- Фабричный метод можно использовать вместе с Итератором, чтобы подклассы коллекций могли создавать подходящие им итераторы.
- Прототип не опирается на наследование, но ему нужна сложная операция инициализации. Фабричный метод наоборот, построен на наследовании, но не требует сложной инициализации.
- Фабричный метод можно рассматривать как частный случай Шаблонного метода. Кроме того, Фабричный метод нередко бывает частью большого класса с Шаблонными методами.

Информационный видеосервис для разработчиков программного обеспечения

ITVDN

IT VIDEO DEVELOPERS NETWORK

TestProvider PROMETRIC TEST CENTER CyberBionic systematics Microsoft Partner Silver Learning Windows Azure Cloud Partner EBA European Business Association