Паттерны проектирования: Паттерн Абстрактной фабрики в TypeScript

Освойте паттерн Абстрактной фабрики и создавайте объекты с легкостью.

Добро пожаловать в серию «Паттерны проектирования в TypeScript», в которой представлены некоторые полезные паттерны проектирования в веб-разработке с использованием TypeScript.

Предыдущие статьи следующие:

Паттерн стратегии в TypeScript

Паттерн цепочки ответственности в TypeScript

Паттерн наблюдателя в TypeScript

Паттерн метода паттерна в TypeScript

Паттерн адаптера в TypeScript

Паттерн фабричного метода в TypeScript

Абстрактный фабричный паттерн в TypeScript

Паттерны проектирования очень важны для веб-разработчиков, и мы можем писать лучший код, освоив их. В этой статье я буду использовать TypeScript, чтобы представить паттерн Абстрактной фабрики.

Паттерн Абстрактной фабрики предоставляет интерфейс для создания партий связанных или взаимозависимых объектов без определения в ходе создания их конкретных свойств (классов).

В паттерне Фабричный метод конкретная фабрика отвечает за производство конкретных продуктов, каждая конкретная фабрика соответствует определенному продукту, и метод у каждой фабрики также уникален. Как правило, в конкретной фабрике существует только один метод создания объектов или группа перезагружаемых методов, но они предназначены только для этой конкретной фабрики. Но иногда нам нужна фабрика, которая может поставлять несколько продуктов, а не один продукт.

На картинке выше я смоделировал процесс покупки автомобиля пользователем. Bytefer заказал SuperX01 на заводе SuperX, и завод изготовил его по модели, соответствующей SuperX01, и доставил его bytefer после завершения производства. Chris1993 заказал SuperX02 на той же фабрике SuperX, и этот же завод изготовил его по образцу, соответствующему SuperX02, и доставила chris1993 после завершения производства.

Давайте посмотрим, как использовать абстрактную фабрику для описания процесса производства определенной модели автомобиля на автомобильном заводе.

Чтобы лучше понять код, давайте сначала изучим соответствующую диаграмму UML:

Паттерн Абстрактная фабрика включает следующие роли:

• Product(Vehicle): абстрактный продукт

• Concrete Product(SuperX01): конкретный продукт

• Factory (SuperXFactory): абстрактная фабрика

• ConcreteFactory(ConcreteSuperXFactory): конкретная фабрика

Определим абстрактный класс Vehicle и два его подкласса SuperX01 и SuperX02 , которые представляют два различных типа транспортных средств.

Теперь определим класс SuperXFactory для представления фабрики по производству автомобилей. Эта абстрактная фабрика содержит абстрактные методы для производства моделей автомобилей SuperX01 и SuperX02.

На основе абстрактного класса SuperXFactory мы определяем заводской класс ConcreteSuperXFactory для производства моделей автомобилей SuperX01 и SuperX02:

После создания заводского класса ConcreteSuperXFactory мы можем «запустить» производство транспортных средств:

В итоге выполнения кода, на терминал можно будет получить следующий результат:

Мы представили паттерн Фабричного метода в предыдущей статье, так в чем же разница между ним и паттерном Абстрактной фабрики?

Самая большая разница между паттерном Абстрактной фабрики и паттерном Фабричного метода заключается в том, что паттерн Фабричного метода нацелен на уникальный (иерархический) продукт. А паттерн Абстрактной фабрики должен уметь обращаться с несколькими видами продуктов, а структура Абстрактной фабрики должна соответствовать свойствам нескольких различных продуктов.

Переходя к созданию объектов как продуктов - когда структура абстрактной фабрики сможет создавать все объекты в семействе продуктов, принадлежащих к различным типам, паттерн Абстрактной фабрики станет проще и эффективнее, чем использование паттерна Фабричного метода.