Factory Patterns - абстрактный заводской узор

Снеш Праджапати

10 октября 2016 г. CPOL

В этой статье мы разберемся с абстрактным шаблоном фабрики. Эта статья является третьей частью серии статей о фабричных шаблонах и продолжением Части 1 (Простой заводской шаблон) и Части 2 (Заводской шаблон метода).

Скачать исходный код - 14 КБ

Вступление

В этой статье мы разберемся с абстрактным шаблоном фабрики. Эта статья является третьей частью серии статей о Factory Patterns и является продолжением Части 1 (Простой Factory Pattern) и Части 2 (Factory Method Pattern) . В первой части мы изучили простой шаблон фабрики, а во второй части мы изучили шаблон метода фабрики, теперь его очередь для абстрактного шаблона фабрики. Если вы не прошли первую и вторую часть, я предлагаю вам сделать это, а затем продолжайте эту статью.

Контуры

Часть 1 - Простой заводской шаблон

Часть 2 - Шаблон заводского метода

Часть 3 - Абстрактный узор фабрики

- Что такое абстрактный узор фабрики
- Когда использовать абстрактный заводской паттерн
- Понять определение на примере
- Проблема с шаблоном абстрактной фабрики

Что такое абстрактный узор фабрики

Абстрактный заводской шаблон является частью книги Gang of Four (GoF) и относится к категории Creational Pattern. Ниже приводится определение абстрактного шаблона фабрики, взятое из книги Gang of Four (GoF)

«Обеспечение интерфейса для создания семейств связанных или зависимых объектов без указания их конкретных классов».

Мы разберем это определение подробно в разделе « *Понять определение на примере* » этой статьи.

Когда использовать абстрактный заводской паттерн

Всякий раз, когда нам нужно создать разные типы связанных объектов (группы / набора), тогда выбор абстрактного фабричного шаблона. Он предоставляет различные виды фабрик. Каждая фабрика создаст определенный вид связанных объектов. Вот почему иногда мы называем шаблон абстрактной фабрики фабрикой фабрик. Simple Factory Pattern и Factory Method Pattern предоставляют только один вид для объектов (в наших примерах все эти объекты относятся к типу IFan).

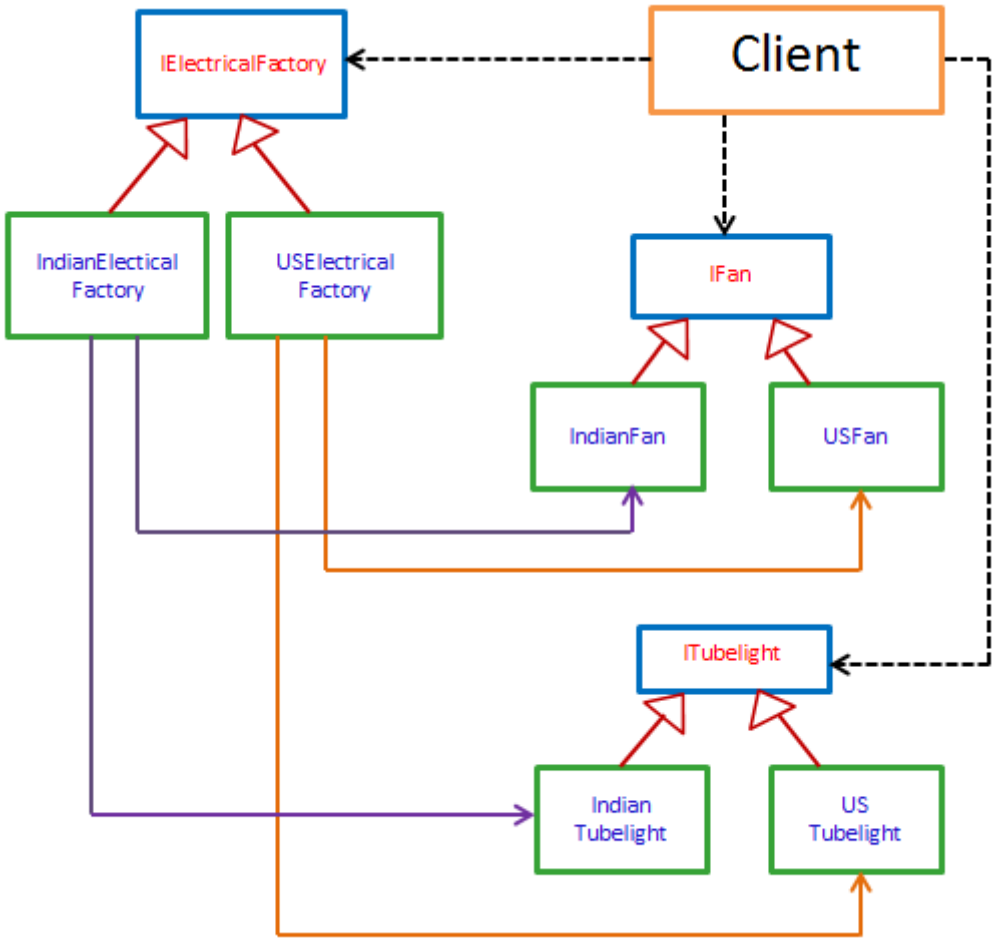
Примечание: абстрактный фабричный шаблон можно использовать вместе с фабричным методом, если это необходимо.

Теперь, кроме вентилятора, если нам нужно создать больше электрического оборудования, такого как Tubelights или Switches, мы выберем Abstract Factory. Чтобы сделать его более понятным, давайте расширим сценарий, в котором нам нужно будет создать различные типы связанных объектов группы (объекты в группе электрического оборудования). Мы продолжим аналогичный пример, который мы рассмотрели во второй части этой серии статей.

Теперь давайте разберемся в расширенном сценарии: существует электротехническая компания, которая производит различное электрическое оборудование, а именно Fan и Tubelight. В настоящее время компания работает только в Индии и называется «Индийская электрическая компания». Теперь та же компания хочет открыть новый филиал в США, а именно компанию US Electrical, которая будет производить электрическое оборудование под названием Fan и Tubelight в соответствии со стандартами США. Итак, у нас есть два типа групп электрического оборудования - это индийское электрическое оборудование и электрическое оборудование США. И связанные объекты в этих группах - Fan и Tubelight (такого электрического оборудования может быть больше).

Понять определение на примере

Давайте разберемся с определением абстрактного шаблона фабрики с помощью вышеупомянутого сценария. Сначала создайте схему проекта вышеупомянутого сценария и попытайтесь соотнести определение с этим сценарием. На схеме ниже показаны необходимые объекты для реализации вышеупомянутого сценария.



*На приведенной выше диаграмме IElectricalFactory предоставляет **клиенту интерфейс для создания семейств связанных или зависимых объектов.** Здесь у нас есть две конкретные реализации этого интерфейса - классы IndianElectricalFactory и USElectricalFactory. Эти два класса производят два разных типа **семейств связанных объектов - Fan и TubeLight.**. IndianFan и IndianTubelight принадлежат к семейству IndianElectricalFactory. Классы USTubelight и USFan, принадлежащие к семейству USElectricalFactory.*

Главный компонент / участники паттерна абстрактная фабрика

- AbstractFactory (IElectricalFactory)
- Бетонный завод (IndianElectricalFactory, USElectricalFactory)
- AbstractProduct (IFan, ITubeLight)
- ConcreteProduct (IndianFan, IndianTubelight, USFan, USTubelight)

Ниже приводится пошаговая реализация вышеупомянутого сценария:

Создайте два интерфейса с именами IFan и ITubelight.

Копировать код

```
interface IFan
{
    void SwithOn();
}

interface ITubelight { }
```

Создайте два конкретных класса, как показано ниже, путем реализации IFan и ITubelight. Реализация интерфейсов будет соответствовать индийскому стандарту электрического оборудования.

Копировать код

```
class IndianFan : IFan { }
```

```
class IndianTubelight : ITubelight { }
```

Создайте интерфейс IElectricalFactory, этот интерфейс является реальной абстрактной фабрикой, которая будет создавать семейства связанных объектов.

Копировать код

```
interface IElectricalFactory
{
    IFan GetFan();
    ITubelight GetTubeLight();
}
```

Создайте класс IndianElecticalFactory и унаследуйте его от интерфейса IElectricalFactory. Это будет реализация двух методов под названием GetFan и GetTubeLight. GetFan возвращает объект класса IndianFan, поскольку интерфейс IFan реализован классом IndianFan. Аналогичным образом GetTubeLight вернет объект класса IndianTubelight.

Копировать код

```
class IndianElectricalFactory : IElectricalFactory
{
    public IFan GetFan()
    {
        return new IndianFan();
    }

    public ITubelight GetTubeLight()
    {
        return new IndianTubelight();
    }
}
```

На данный момент класс IndianElecticalFactory готов к созданию IndianFan и IndianTubelight. Теперь посмотрим, как клиент может их создавать.

Копировать код

```
static void Main(string[] args)
{
    IElectricalFactory electricalFactory = new IndianElectricalFactory();
    IFan fan = electricalFactory.GetFan();
    fan.SwithOn();
    Console.ReadKey();
}
```

Как мы обсуждали выше, компания хочет создать новую электрическую компанию в США, и ее название будет USElectricalFactory. В том же приложении мы добавим два новых класса под названием USFan и USTubelight, реализовав необходимые интерфейсы. Здесь реализация интерфейсов IFan и ITubeLight будет соответствовать стандарту электрического оборудования США.

Копировать код

```
class USFan : IFan { }

class USTubelight : ITubelight { }
```

Наконец, создайте класс USElectricalFactory, который также унаследован от IElectricalFactory.

Копировать код

```
class USElectricalFactory : IElectricalFactory
{
    public IFan GetFan()
    {
        return new USFan();
    }
```

```
    public ITubelight GetTubeLight()
    {
        return new USTubelight();
    }
}
```

Теперь, если клиент хочет получить оборудование USElectrical, это можно сделать, просто сделав одно изменение в клиентском коде, как показано ниже:

Копировать код

```
//IElectricalFactory electricalFactory = new IndianElectricalFactory();
IElectricalFactory electricalFactory = new USElectricalFactory();
```

Проблема с шаблоном абстрактной фабрики

Проблема может возникнуть только в том случае, если мы изменим интерфейс, предоставляемый самой Abstract Factory. Это редкость, и каждая программа, которая следует философии проектирования « *Программа для интерфейса, а не реализация* », страдает от этой проблемы. Та же проблема обсуждается на этой странице stackoverflow . В нашем случае, если в интерфейсе IElectricalFactory происходит какое-либо изменение, все фабрики должны быть изменены.

Заключение

В этой статье мы познакомились с шаблоном абстрактной фабрики и его использованием. Мы поняли контекст абстрактного шаблона фабрики и то, как его использовать для повышения удобства сопровождения приложения. Спасибо за прочтение. Мы будем рады вашим комментариям и предложениям по улучшению.

использованная литература

- Абстрактный узор фабрики
- Абстрактная фабрика против фабричного метода
- Абстрактная фабрика

Лицензия

Делиться

об авторе

Снеш Праджапати
Разработчик программного обеспечения
Индия

Я разработчик программного обеспечения, работающий над технологиями Microsoft. Меня интересует изучение и распространение новейших технологий.

Комментарии и обсуждения

Вы должны войти в систему, чтобы использовать эту доску сообщений.

Search Comments

Первая Предыдущая Следующая

Абстрактный заводской шаблон и заводской шаблон выглядят такими же, как в соответствии с вашим кодом
vishnu_cs47 17-фев-19 23:58

Исправление
Шив Ратан 25-фев-18 4:31

Хороший
Prdissa 23 января 17 18:17

Превосходно!
станино 7-ноя-16 9:41

Re: Отлично!
Snesh Prajapati 7-Nov-16 15:05

Завод Дизайн Патерн
DeyChandan 12-Oct-16 6:06

Re: Патерн Factory Design
Snesh Prajapati 12-Oct-16 16:37

Завод Дизайн Патерн
DeyChandan 12-Oct-16 6:06

Re: Патерн Factory Design
Snesh Prajapati 12-Oct-16 16:37

Лучшая статья о заводских выкройках, которую я когда-либо читал
Patrick Skelton 10-Oct-16 2:12

Re: Лучшая статья о заводских выкройках, которую я когда-либо читал
Snesh Prajapati 10-Oct-16 2:27

Хорошее объяснение
vbalajich **10-Oct-16 0:46**

Re: Хорошее объяснение
Snesh Prajapati 10-Oct-16 1:14

Мой голос за 5
Shambhoo kumar **9-Oct-16 20:26**

Re: Мой голос за 5
Snesh Prajapati 9-Oct-16 21:33

Re: Мой голос за 5
Shambhoo kumar 10-Oct-16 19:35

Обновить 1

Общие Новости Предложение Вопрос Ошибка Ответ Шутка Похвала Rant
Администратор

Используйте Ctrl + Left / Right для переключения сообщений, Ctrl + Up / Down для переключения цепочек, Ctrl + Shift + Left / Right для переключения страниц.