北京航空航天大學
BEIHANG UNIVERSITY

软件体系结构第四次作业

——设计模式

姓　　名:　　姬轶

学　　号:　　ZY2006109

北京航空航天大学

2021 年 05 月

工厂方法模式

在大三实习时参与了商城系统的设计，采用 Spring Boot 进行开发，其中购买方式模块、支付模块采用了工厂方法模式，故此次作业详细介绍 23 种设计模式中的工厂模式。

1.1. 模式的结构与实现

工厂方法模式是对简单工厂模式的进一步抽象化，其好处是可以使系统在不修改原来代码的情况下引进新的产品，即满足开闭原则。

意图：

定义一个创建对象的接口，让其子类自己决定实例化哪一个工厂类，工厂模式使其创建过程延迟到子类进行。

主要解决：

主要解决接口选择的问题。

何时使用：

我们明确地计划不同条件下创建不同实例时。

如何解决：

让其子类实现工厂接口，返回的也是一个抽象的产品。

关键代码：

创建过程在其子类执行。

优点：

- 用户只需要知道具体工厂的名称就可得到所要的产品，无须知道产品的具体创建过程。
- 灵活性增强，对于新产品的创建，只需多写一个相应的工厂类。
- 典型的解耦框架。高层模块只需要知道产品的抽象类，无须关心其他实现类，满足迪米特法则、依赖倒置原则和里氏替换原则。

缺点：

- 类的个数容易过多，增加复杂度
- 增加了系统的抽象性和理解难度
- 抽象产品只能生产一种产品，此弊端可使用抽象工厂模式解决。

模式的结构：

工厂方法模式由抽象工厂、具体工厂、抽象产品和具体产品等 4 个要素构成。本节来分析其基本结构和实现方法。工厂方法模式的主要角色如下。

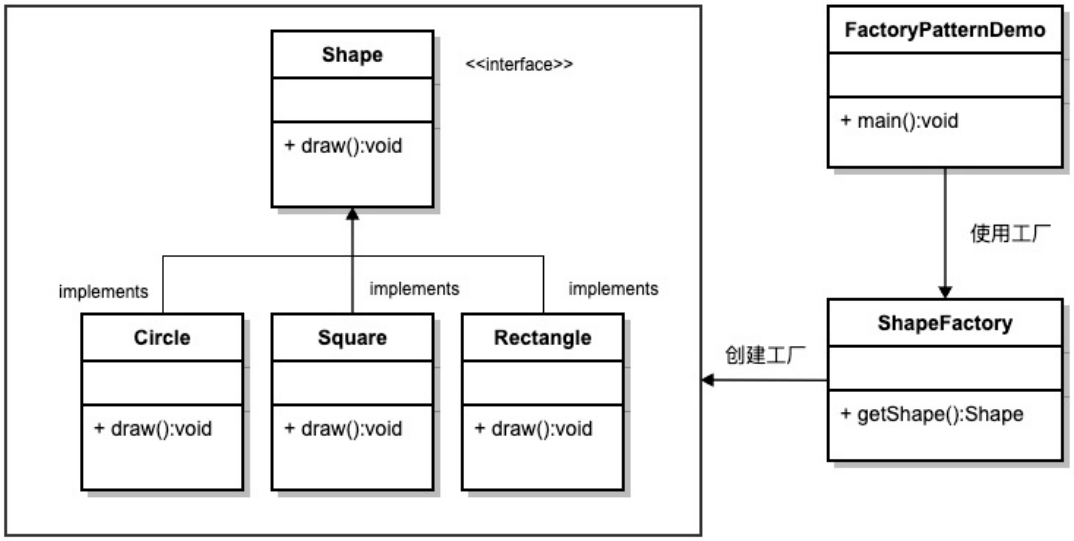
- 抽象工厂（Abstract Factory）：提供了创建产品的接口，调用者通过它访问具体工厂的工厂方法 newProduct() 来创建产品。
- 具体工厂（ConcreteFactory）：主要是实现抽象工厂中的抽象方法，完

成具体产品的创建。

- 抽象产品（Product）：定义了产品的规范，描述了产品的主要特性和功能。
- 具体产品（ConcreteProduct）：实现了抽象产品角色所定义的接口，由具体工厂来创建，它同具体工厂之间一一对应。

注意事项：

作为一种创建类模式，在任何需要生成复杂对象的地方，都可以使用工厂方法模式。有一点需要注意的地方就是复杂对象适合使用工厂模式，而简单对象，特别是只需要通过 new 就可以完成创建的对象，无需使用工厂模式。如果使用工厂模式，就需要引入一个工厂类，会增加系统的复杂度。



图一 工厂模式简单样例

1.2. 具体场景

1.2.1. 场景 1：购买方式模块

在购买方式上，存在正常购买、秒杀、预购、团购、周期购五种方式。当然，我们可以通过将购买方式全部注入到容器里，根据 bean 去容器中获取相应的内容，此时代码灵活性极差。

在这种情况下，我们只要再注入一个工厂作为购买工厂，每次去购买工厂获取自己想要的购买方式，这样我们就可以像之前那样获取到自己想要的内容。具体流程如下：

1. 定义下单接口
2. 实现下单接口
   a) 秒杀订单

b) 普通订单
c) 预售订单
d) 团购订单
e) 周期购订单

3. 创建工厂方法的工厂
4. 调用

1.2.2. 场景 2：支付模块

在支付方式上，存在微信支付、支付宝支付、余额支付三种方式。

对于上述案例的需求，我们使用工厂方法模式，Pay 是一个抽象的工厂类，然后定义了不同支付方式的类：微信支付工厂类（WeChatPay），支付宝支付工厂类（AliPay），余额支付工厂类（BalancePay），这三个支付工厂类分别去支付相应支付方式对应的的订单。相应流程如上场景所示。