设计模式

1. 设计模式分类

创建型模式：用于描述“如何创建对象”，主要特点是将对象的创建和使用分离。

结构型模式：如何将类和对象按照某种布局构成更大的结构。

行为型模式：描述类和对象之间协作共同完成单个对象无法单独完成的任务，以及如何分配职责。

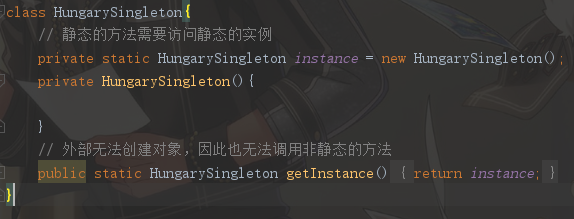
1. 创建型模式

①单例模式

涉及到一个单一的类，该类负责创建自己的对象，同时确保只有单个对象被创建。该类提供了一种访问其唯一对象的方式，不需要实例化该类。

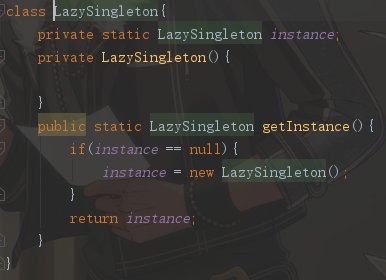
1. 饿汉式

在类初始化时就实例化。

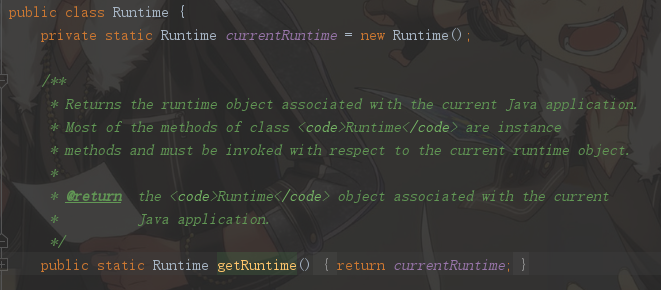


1. 懒汉式

在首次使用该对象时才进行实例化。



1. 具体实现：JDK Runtime



②工厂模式

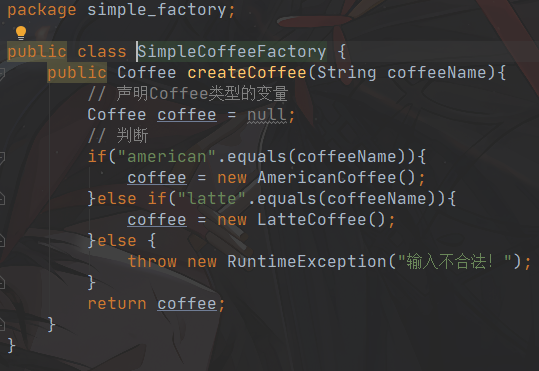
如果创建对象时直接new一个对象，就会对该对象耦合严重，如果要更换对象，所有new对象的地方都需要修改一遍。

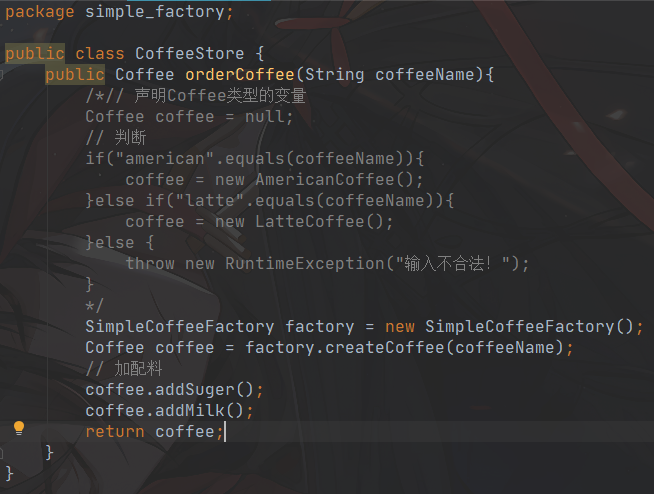
如果用工厂来生产对象，当需要更换对象时，只需要在工厂里进行修改。

最大的优点：解耦。

1. 简单工厂

组成：抽象产品（定义产品的规范），具体产品（抽象产品的实现或者子类），具体工厂（获取产品）





现在CoffeeStore不生产具体的咖啡，直接使用简单工厂生产的产品，解除了依赖。

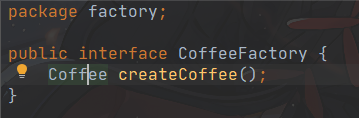
把对象的创建和业务逻辑层分离，避免了修改客户代码（Coffee Store）。

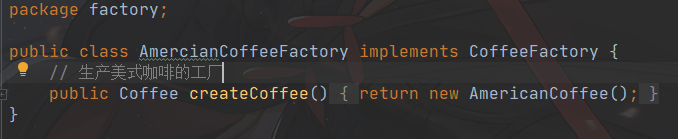
缺点：仍然不满足开放-封闭原则，当需要添加一种新的产品时，需要修改简单工厂类。

1. 工厂模式

定义一个用于创建对象的接口，让子类决定实例化哪个产品对象。

组成：抽象产品，具体产品，**抽象工厂**（创建对象的接口），具体工厂（生产具体产品）。





满足开放-封闭原则：当需要生产新的产品时，只需要创建一个新的具体工厂，实现抽象工厂接口即可，不需要修改工厂类。

缺点：每增加一个新的产品都必须增加一个新的具体工厂和一个新的具体产品，增加了系统的复杂度。

1. 抽象工厂模式

抽象工厂模式考虑多等级产品的生产（甜品店不止卖不同种类的咖啡，还卖各种甜品）。

产品族：同一个具体工厂生产的，位于不同等级（不同种类）的一组产品。

组成：抽象产品，具体产品，抽象工厂（包含多个创建产品的方法，可以生产同族但不同等级的产品），具体工厂（实现抽象工厂的多个方法）。