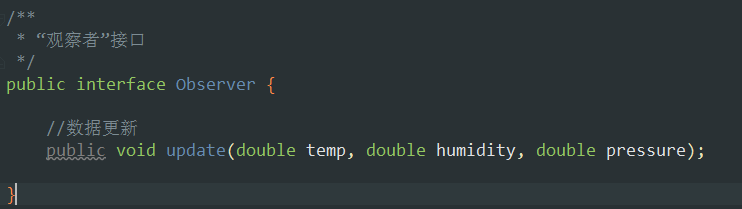
**《**Head Frist设计模式**》笔记:**

**一.观察者模式：**

**观察者模式定义了对象之间的一对多依赖，这样一来，当一个对象（该对象为主题）改变状态时，它的所有依赖者（观察者）都会收到通知并自动更新。比如报纸订阅服务，每当有新报纸（报纸为主题）时，订阅者（订阅者为观察者）会自动收到新报纸，订阅者也可以随时取消订阅，未订阅者也可以随时进行订阅。**

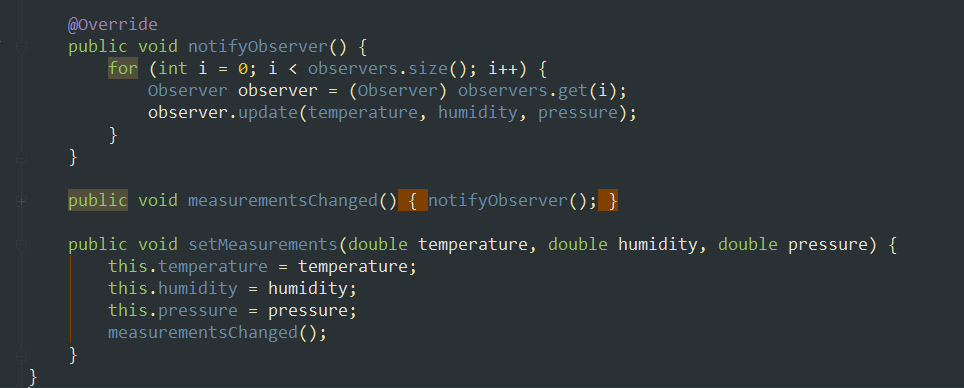
**典型：Tomcat的Listener。**

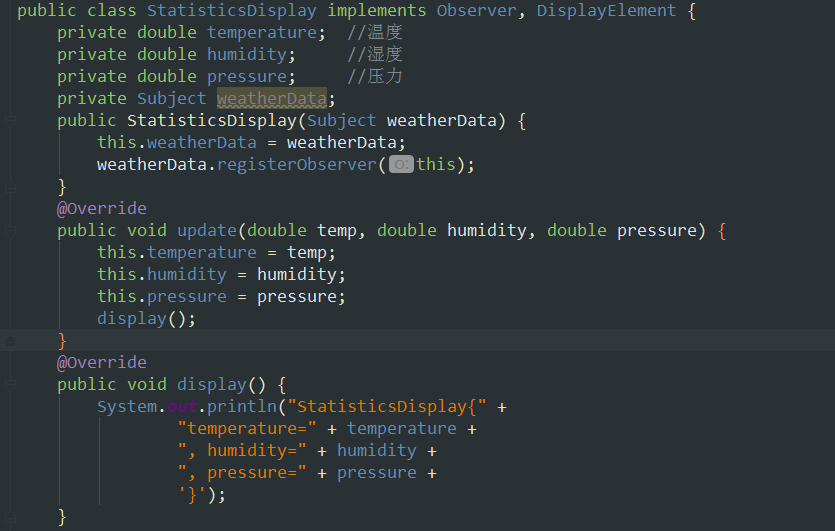


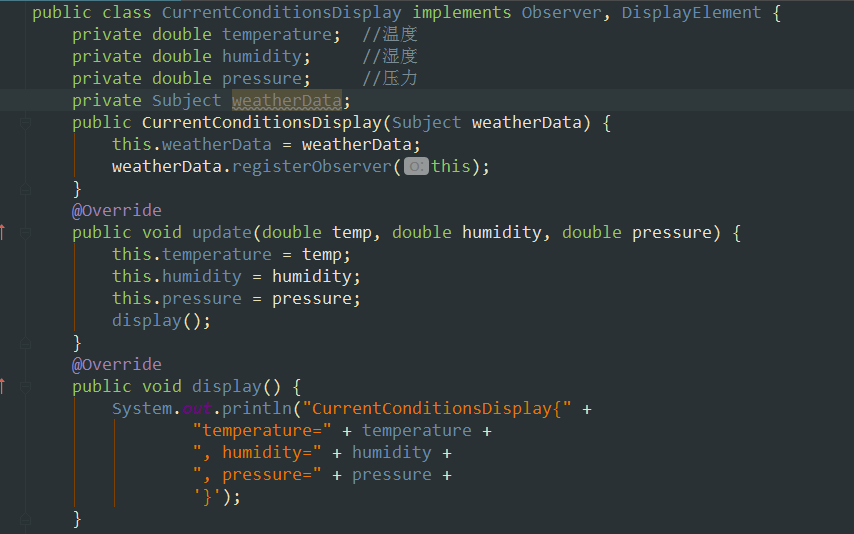


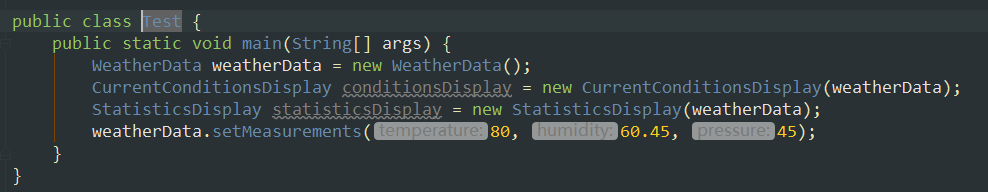




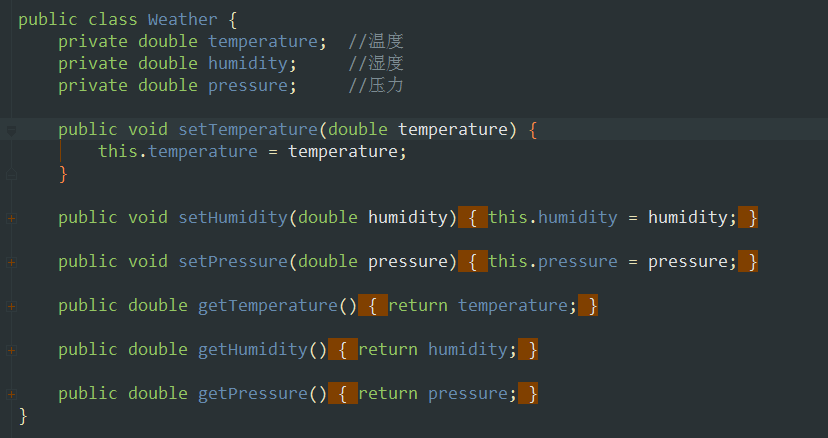


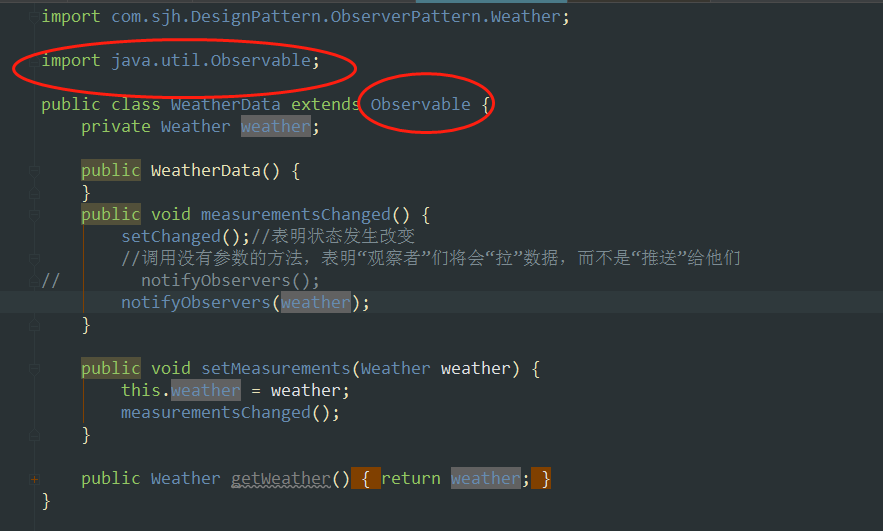




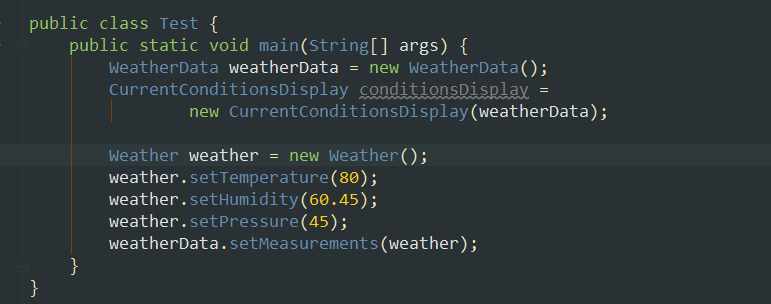


**Java内置观察者模式：**







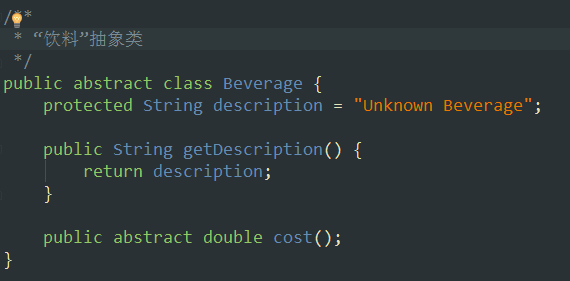


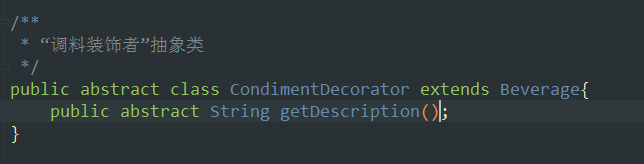
**二.装饰者模式：**

**装饰者模式动态地将责任附加到对象上，若要扩展功能，装饰者提供了比继承更有弹性的替代方案。**

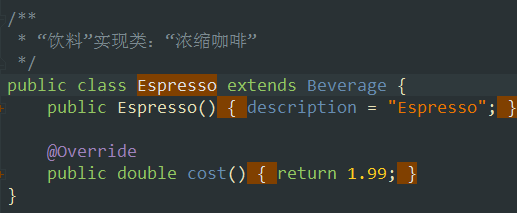
开放-关闭原则：类应该对扩展开放，对修改关闭。

**典型：JDK的IO流。**















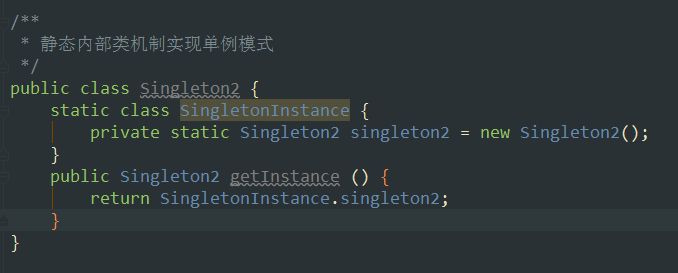


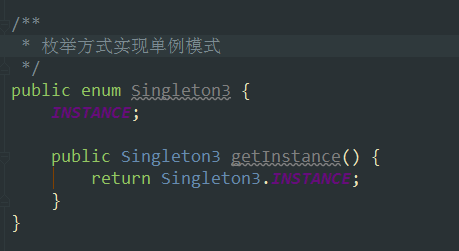
**三.单件模式：**

单件模式确保一个类只有一个实例，并提供一个全局访问点。

**典型：Spring的Bean。**



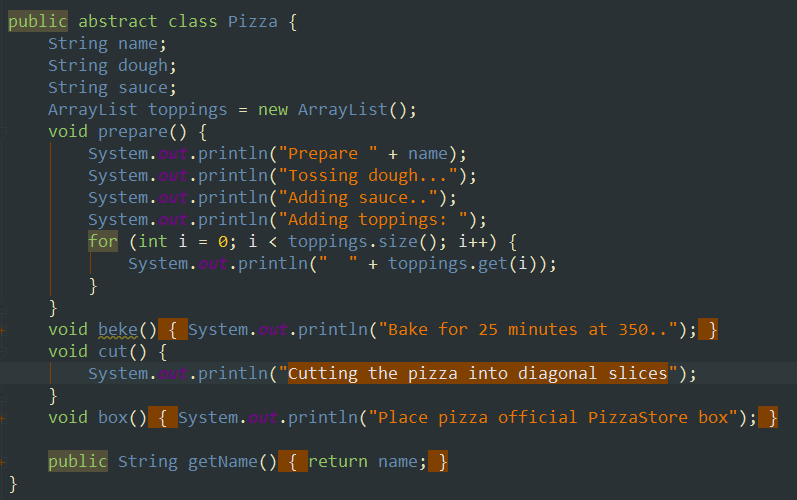


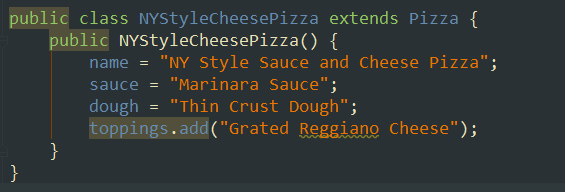


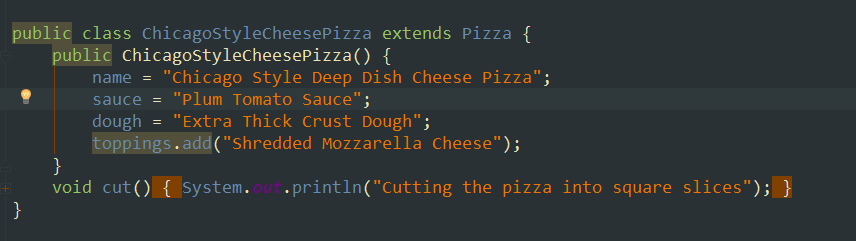
**四.工厂模式：**

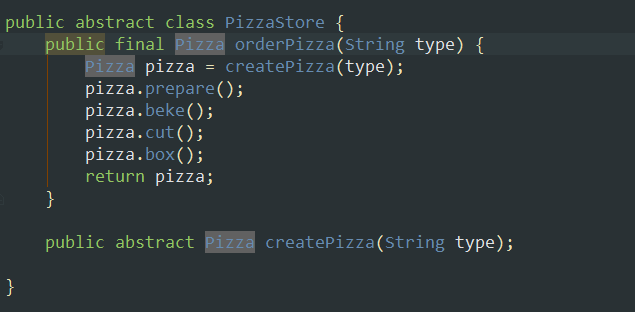
工厂方法模式定义了一个创建对象的接口，但由子类决定要实例化的类是哪一个。工厂方法让类的实例化推迟到子类。

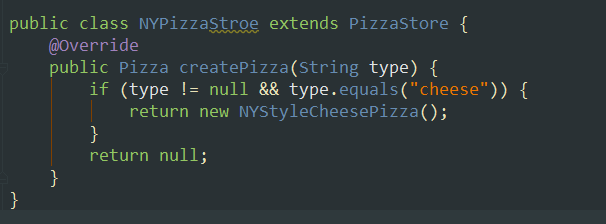
**典型：Spring的BeanFactory、MyBatis的SqlSessionFactory。**

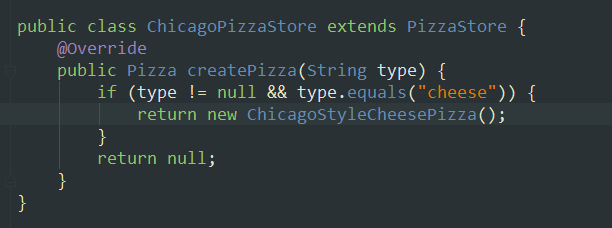


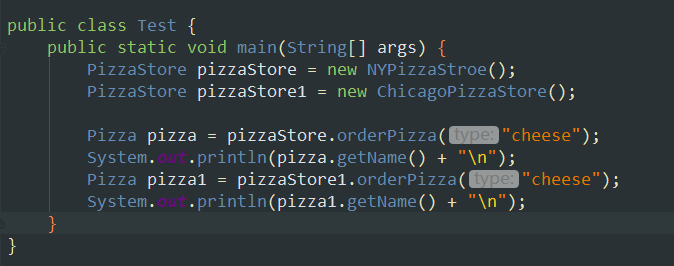




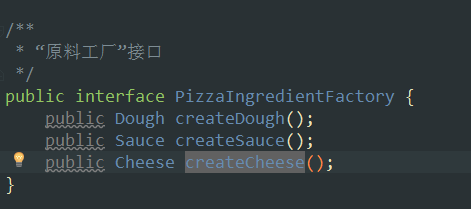


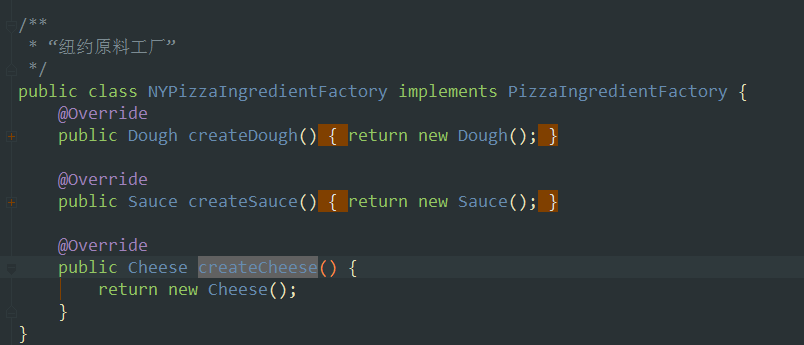


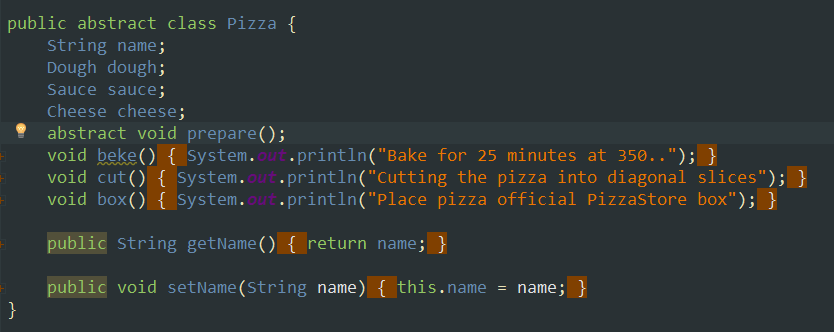


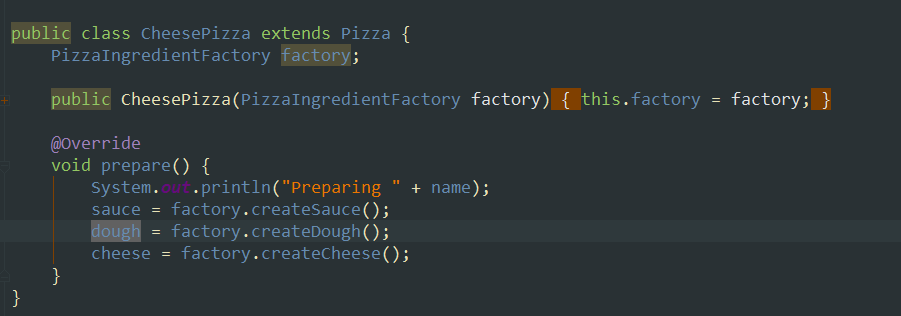


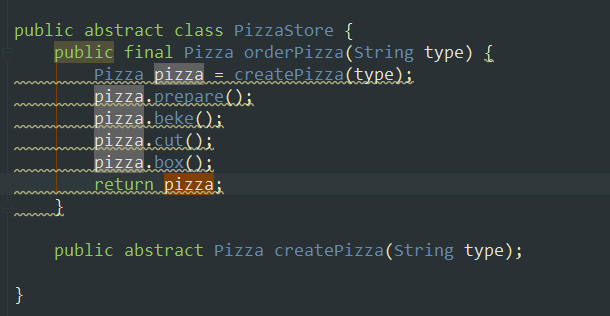
抽象工厂模式提供了一个接口，用于创建相关或依赖对象的家族，而不需要明确指定具体类。

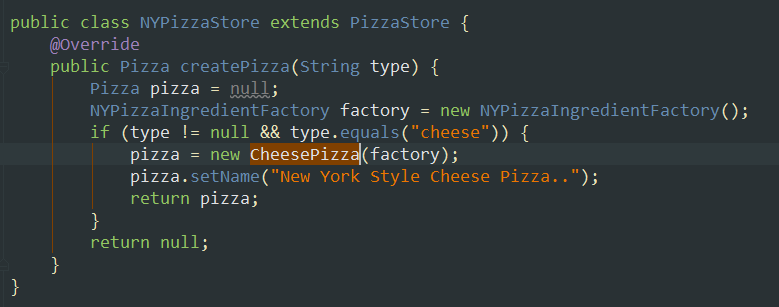


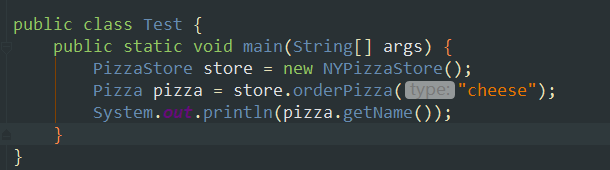






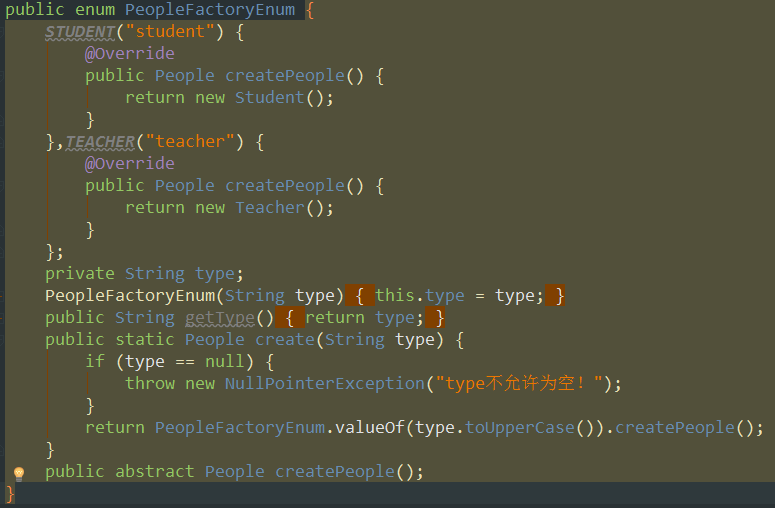






依赖倒置原则：要依赖抽象，不要依赖具体类。

枚举实现工厂模式：



**五.命令模式：**

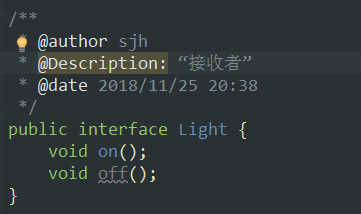
命令模式将“请求”封装成对象，以便使用不同的请求、队列或者日志来参数化其他对象。

当需要将发出请求的对象和执行请求的对象解耦的时候，使用命令模式。

**典型：Tomcat对请求的处理。**

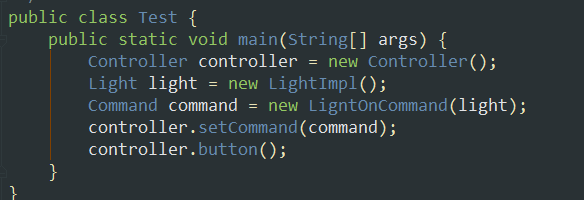






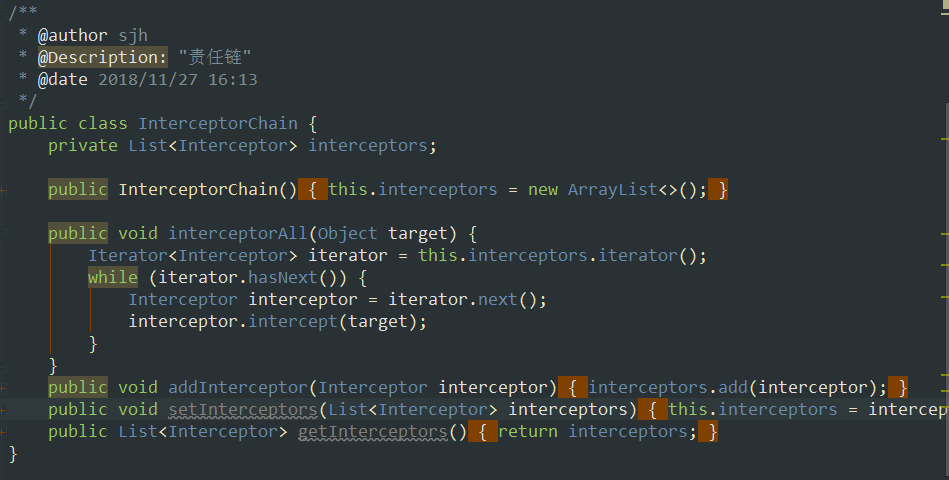


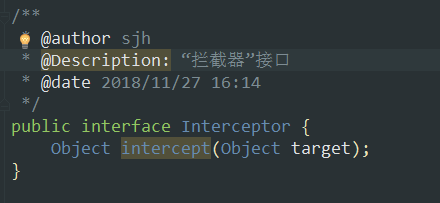


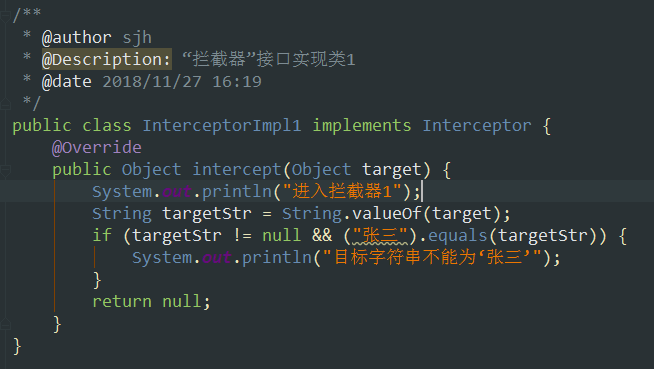


**六.责任链模式：**

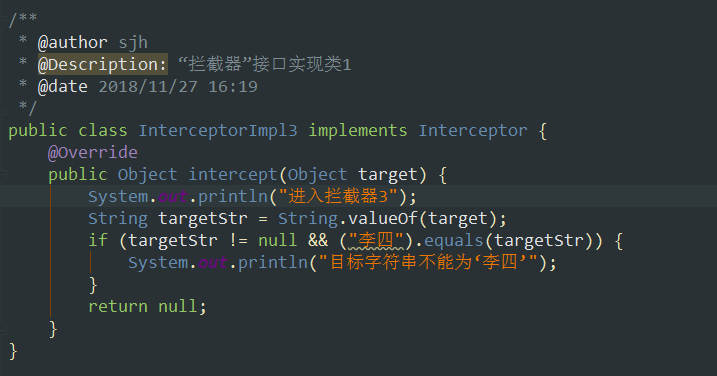
责任链模式为某个请求创建一个对象链，每个对象依次检查此请求，并对其进行处理，或者将它传到下一个对象。

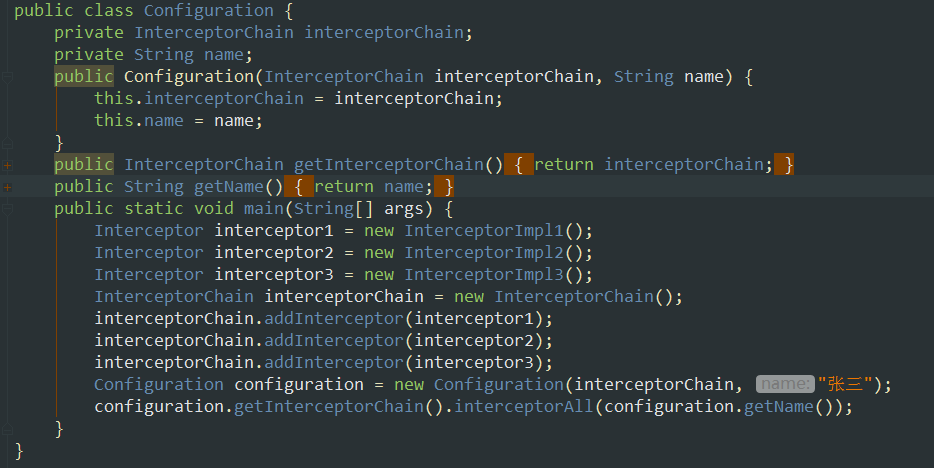




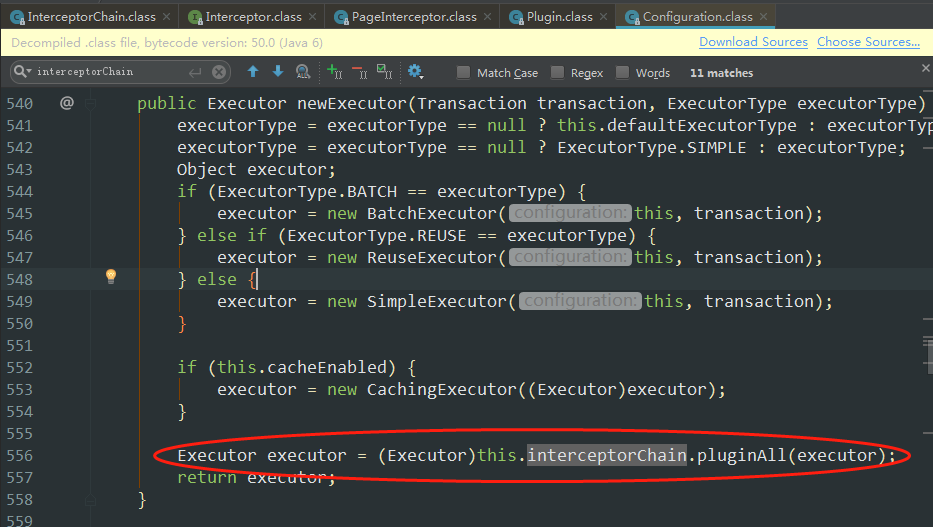


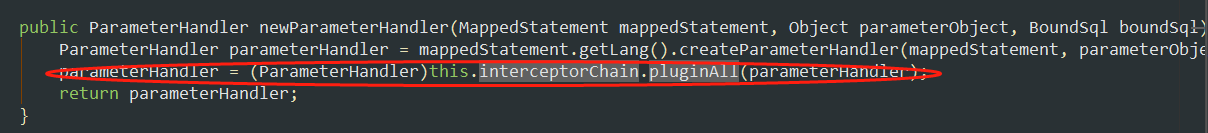


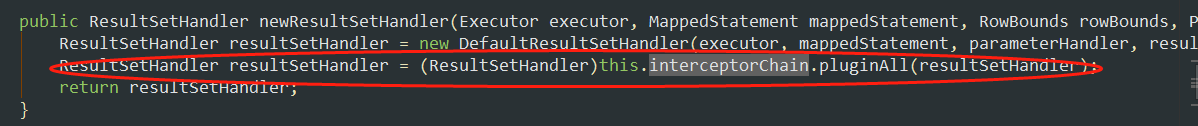


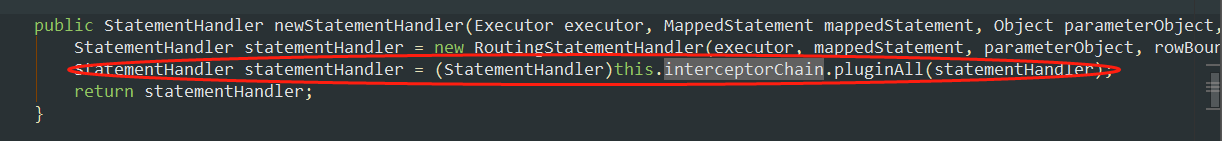


**典型示例（MyBatis插件机制）：**





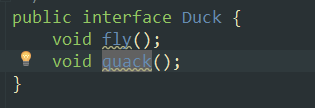


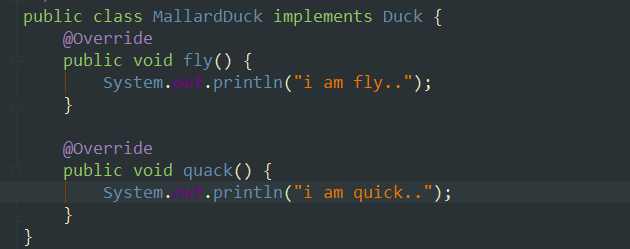


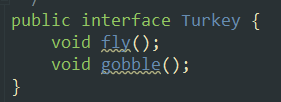
**七.适配器模式和外观模式：**

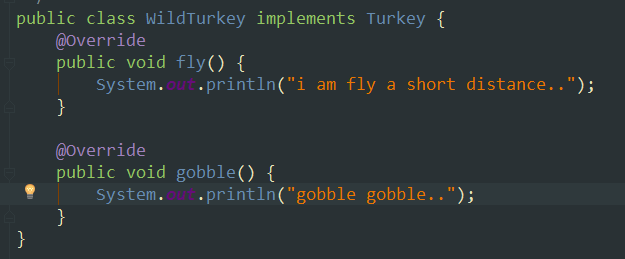
一、适配器模式：将一个类的接口转换成客户期望的另一个接口，适配器让原本不兼容的类可以合作无间。可以让客户与被适配者解耦。

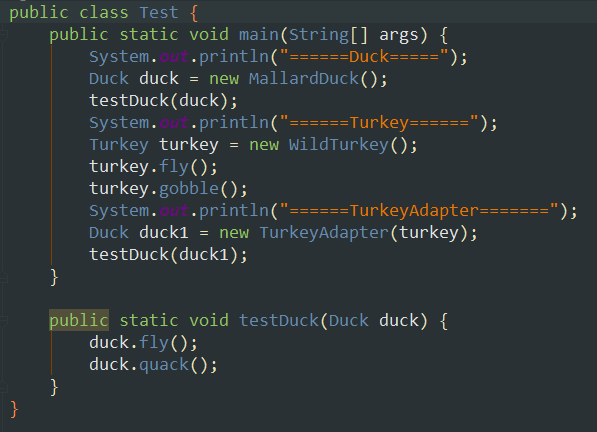












二、外观模式：