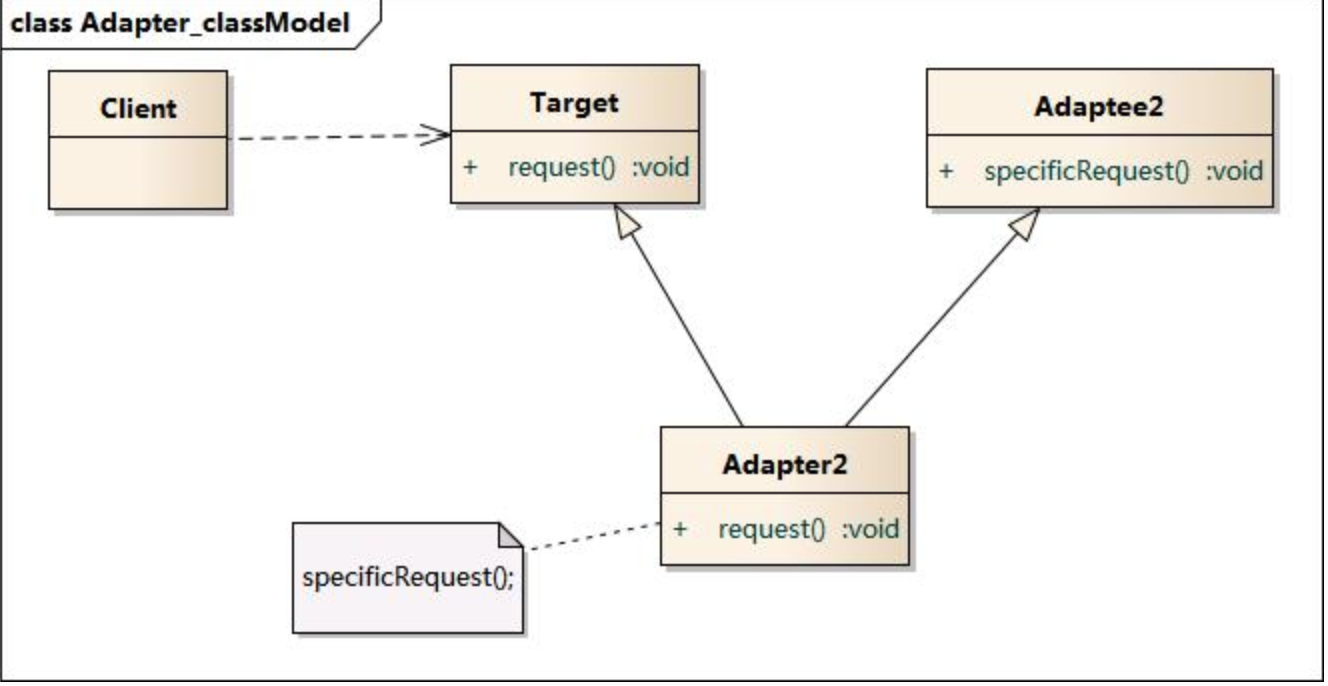
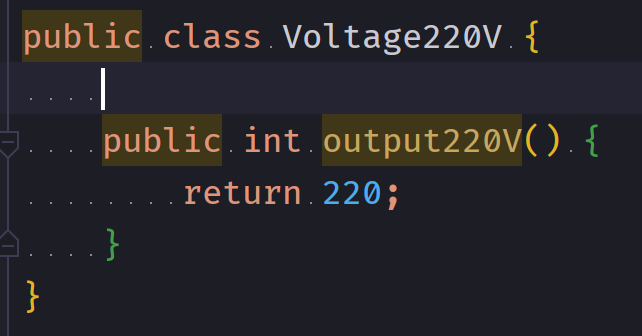
1. 适配器模式(Adapter Pattern)：将一个类的接口转换成客户期望的另一个接口，让原本不兼容的类可以合作无间(类似于插头转换)。(使原本由于接口不兼容不能一起工作的类可以一起工作)。适配器模式的主要目的是**兼容性**，其别名为包装器(Wrapper)，属于**结构形**模式。主要分为三类：类适配器、对象适配器、接口适配器



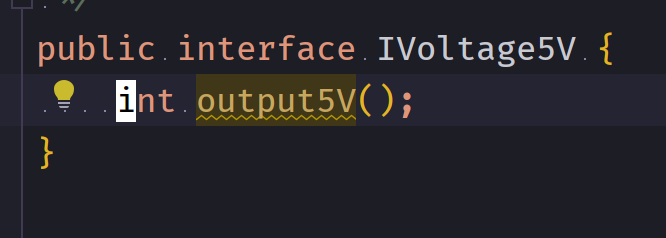
1. 类适配器
2. 基本介绍: Adapter2类，通过继承Adaptee2类，实现Target接口，完成adaptee->target的适配。主要从类的继承上解决问题
3. 类图



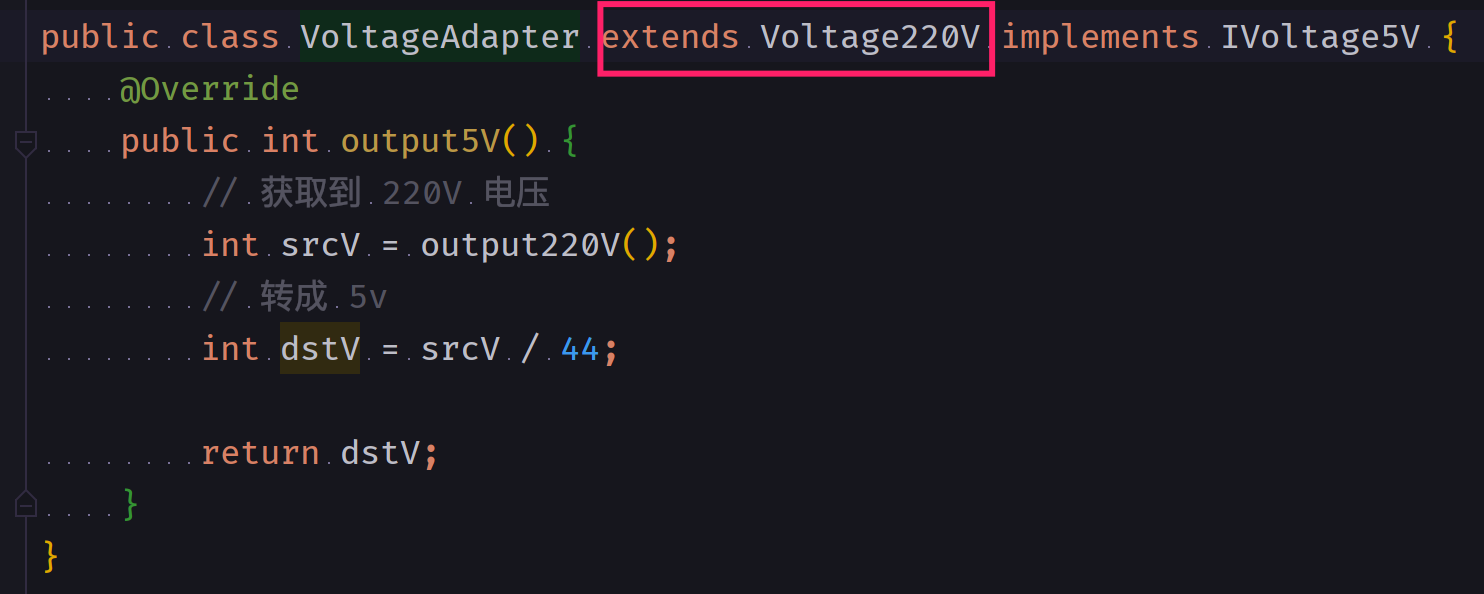
1. 实例，以生活中充电器的例子来讲解适配器，充电器本身相当于Adapter，220V 交流电相当于Adaptee (即被适配者)，我们的Target(即目标)是5V直流电
   1. 被适配的类Adaptee



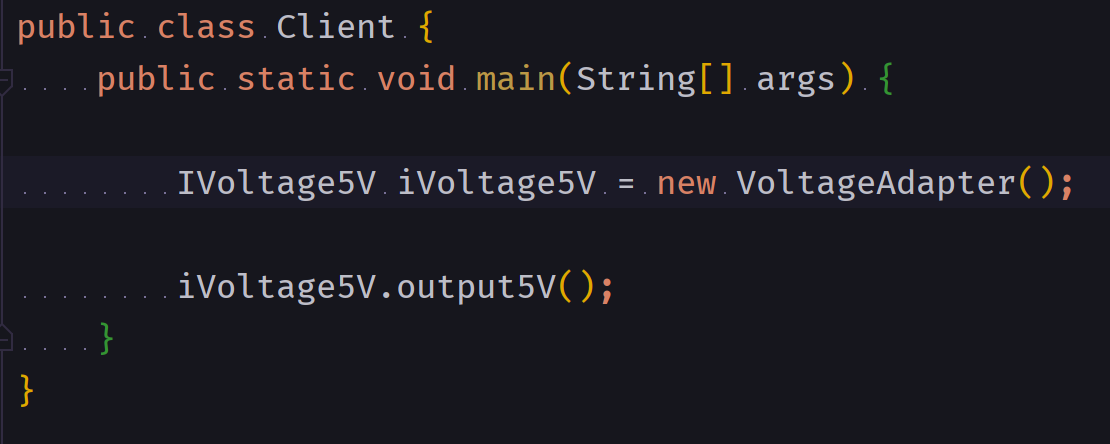
* 1. 适配接口Target



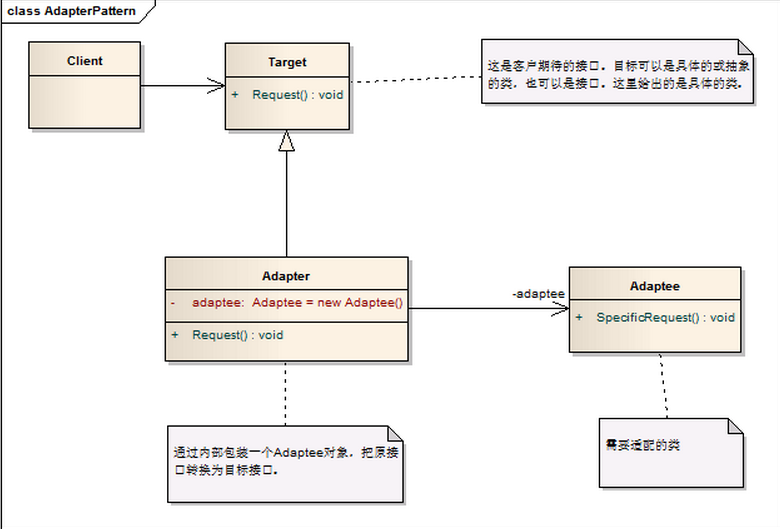
* 1. 适配器Adapter



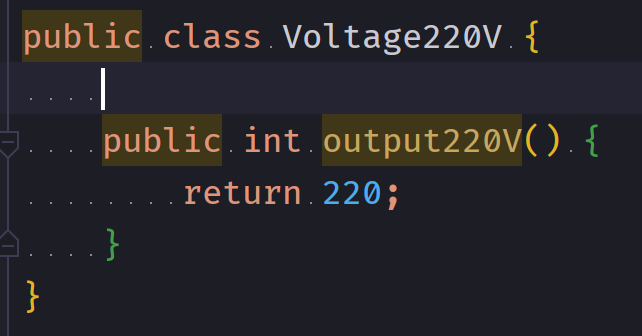
* 1. 使用Client



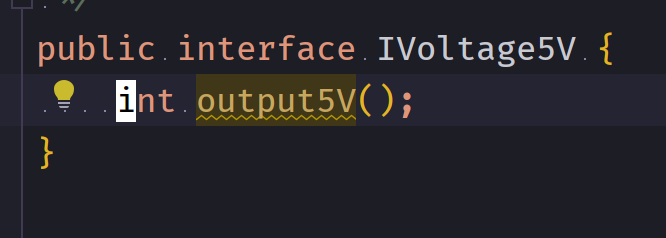
1. 类适配器模式注意事项
   1. Java是单继承机制，所以类适配器需要继承Adaptee类这一点算是一个缺点，因为这要求Target必须是接口，有一定局限性
   2. Adaptee类的方法在Adapter中都会暴露出来，也增加了使用的成本
   3. 由于Adapter继承了Adaptee类，所以它可以根据需求重写Adaptee类的方法，使得Adapter的灵活性增强了
2. 对象适配器
   1. 基本思路和类的适配器模式相同，只是将Adapter类作修改，不是继承Adaptee类，而是持有Adaptee类的实例，以解决兼容性的问题。即：持有 Adaptee类，实现Target 类接口，完成adaptee->target的适配。对象适配器模式是适配器模式常用的一种
   2. 根据“合成复用原则”，在系统中尽量使用关联关系(聚合)来替代继承关系
   3. 类图



* 1. 实例，以生活中充电器的例子来讲解适配器，充电器本身相当于Adapter，220V 交流电相当于Adaptee (即被适配者)，我们的Target(即目标)是5V直流电
     1. 被适配的类Adaptee



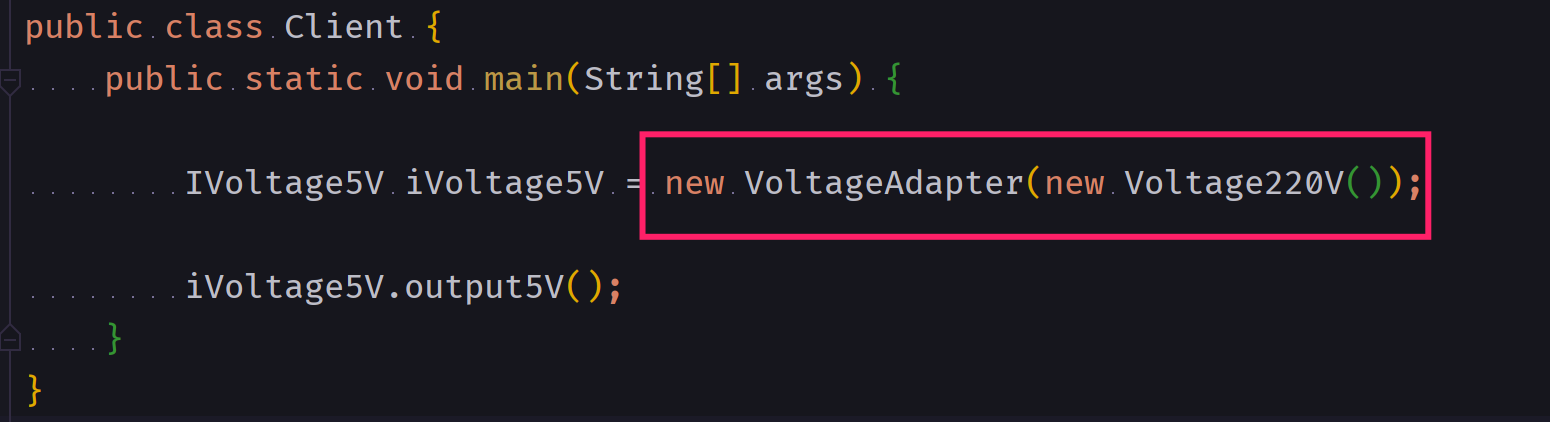
* + 1. 适配接口Target



* + 1. 适配器Adapter

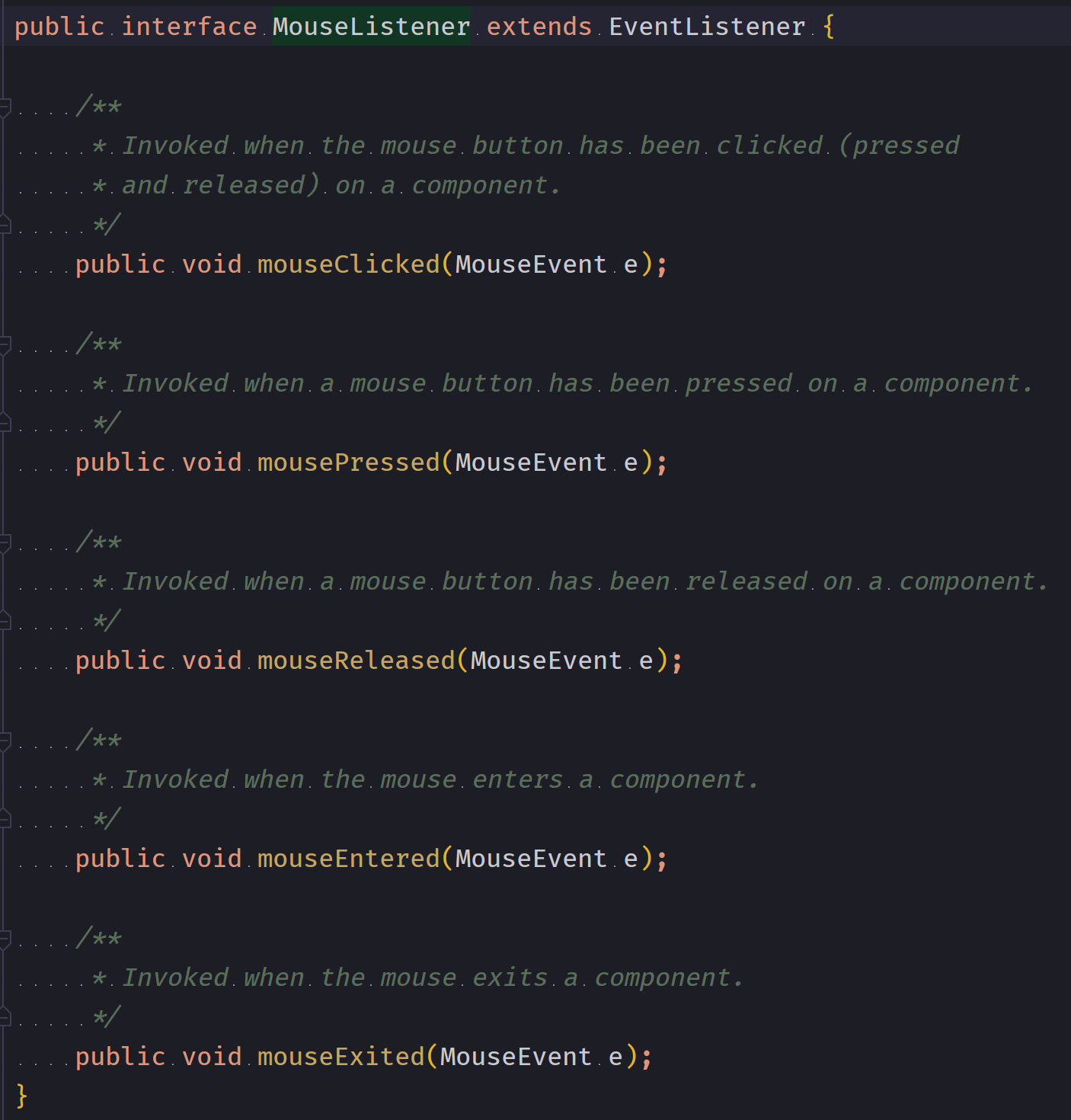


* + 1. 使用Client

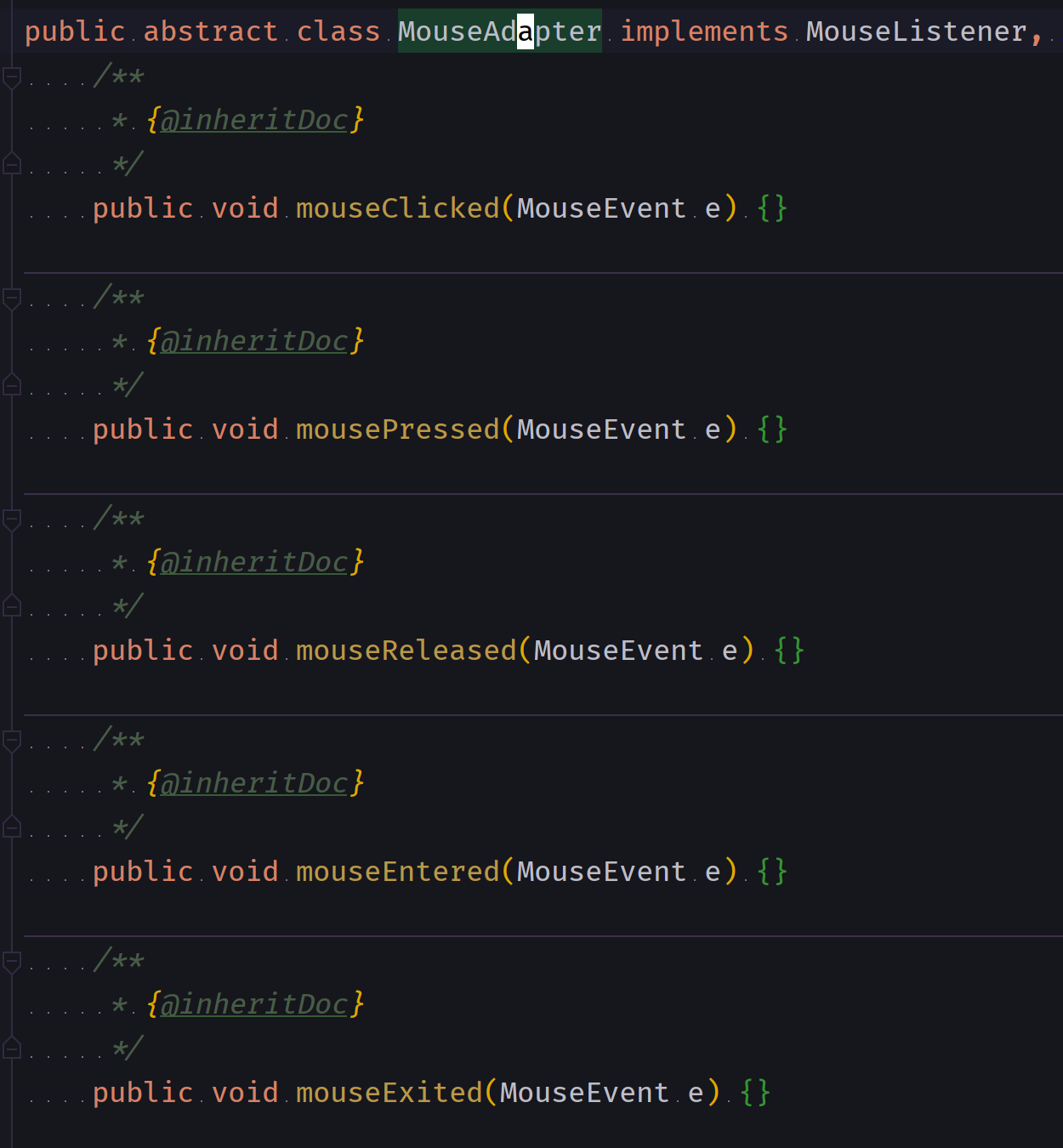


* 1. 对象适配器模式的三个特点
  2. 适配器对象实现原有接口(Duck)
  3. 适配器对象组合一个实现新接口(Turkey)的对象（这个对象也可以不实现一个接口，只是一个单纯的对象）
  4. 对适配器原有接口方法的调用被委托给新接口的实例的特定方法
  5. 常见场景:
  6. java.io.InputStreamReader#InputStreamReader(java.io.InputStream)
  7. java.io.OutputStreamWriter#OutputStreamWriter(java.io.OutputStream)

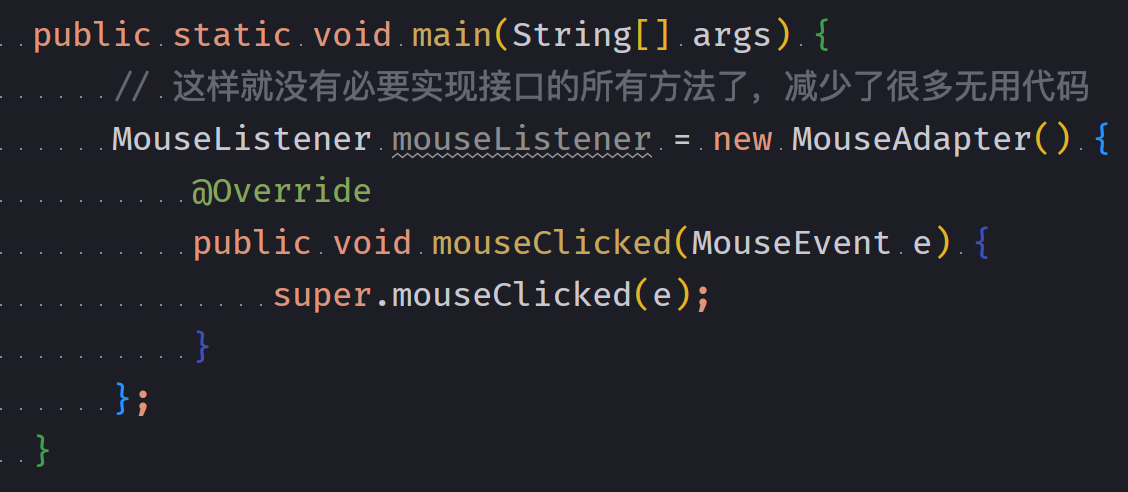
1. 接口适配器
   1. 核心思路: 当**不需要全部实现接口提供的方法时**，可先设计一个抽象类实现接口并为该接口中每个方法提供一个默认实现(空方法)，那么该抽象类的子类可**有选择**地覆盖父类的某些方法来实现需求。
   2. 适用于一个接口不想使用其所有的方法的情况，可以让用户更加方便的使用该接口，减少很多无用代码
   3. 实例
      1. 接口java.awt.event.MouseListener



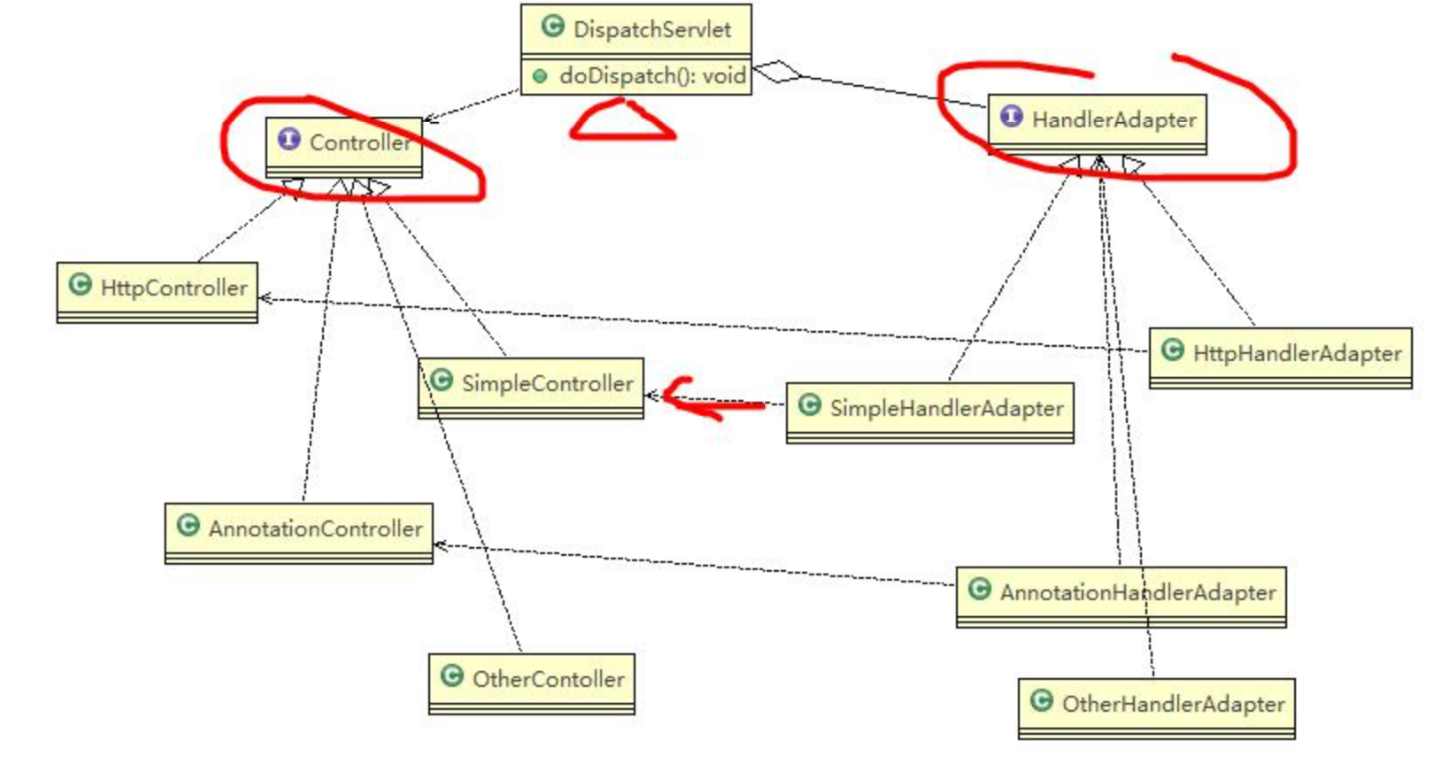
* + 1. 接口适配器java.awt.event.MouseAdapter



* + 1. 使用Client



1. SpringMVC中的org.springframework.web.servlet.HandlerAdapter，就使用了适配器模式
   1. 使用HandlerAdapter原因: 处理器的类型不同，有多重实现方式，那么调用方式就不是确定的，如果需要直接调用Controller的方法，需要调用的时候就得不断使用 if else来进行判断是哪一种子类然后执行。那么如果后面要扩展Controller，就得修改原来的代码，这样违背了OCP原则



* 1. 不同的Controller实现





* 1. org.springframework.web.servlet.DispatcherServlet#doDispatch



1. 适配器模式注意事项/细节
   1. 三种命名方式，是以适配的类Adaptee是以怎样的形式给到适配器Adapter来命名的
      1. 类适配器: 以类的形式给到，在Adapter里，就是将Adaptee当做**类继承**
      2. 对象适配器：以对象的形式给到，在Adapter里，就是将Adaptee当做**对象持有**
      3. 接口适配器：以接口的形式给到，在Adapter里，就是将Adaptee当做**接口实现**
   2. 适配器模式最大的作用是将原本不兼容的接口融合在一起工作
   3. 实际开发中，实现起来不拘泥于以上三种类形式