装饰者模式

为何需要这个东西？

问题：假设，我们有一个基础功能 A，希望A可以具备若干个额外的特性：f1,f2。且这三种特性可以任意组合。如何实现？

对于功能扩展我们可以通过继承来实现，我们可以为A写3个子类：A_f1、A_f2，这3个子类中分别实现了f1,f2的特性。

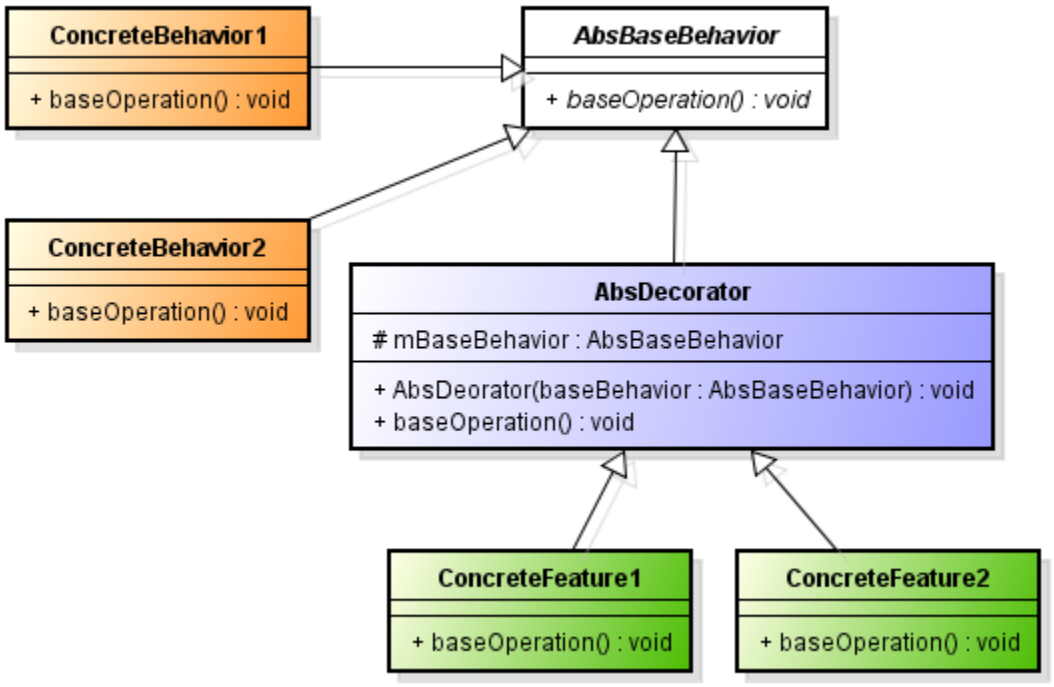
对于同时带有 f1和f2特性的情况，还得再添加一个类 A_f1_f2，随着额外特性的数量增多，我们要新增的类数量呈指数级上涨。

显然对于这种情况通过继承来扩充类功能不是很好的选择。这个时候装饰者模式就派上用场了。

定义

装饰者模式：在不改变原类文件以及不使用继承的情况下，动态地将责任附加到对象上，从而实现动态拓展一个对象的功能。它是通过创建一个包装对象，也就是装饰来包裹真实的对象。

类图

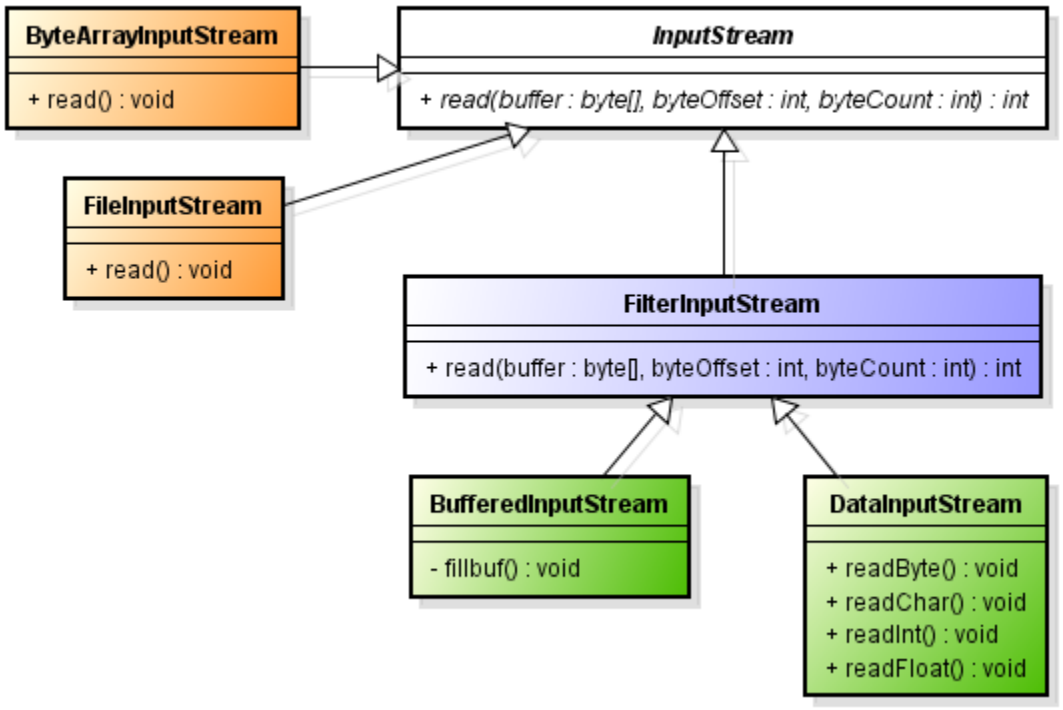


使用

通常情况下我们是这样使用的： ConcreteFeature1 a = new ConcreteFeature1(new ConcreteFeature2(baseBehavior));

应用举例

Java 中输入/输出流，首先来看下类图（是不是和上面的结构很相似？）：



多级缓存的简单嵌套： new BufferedInputStream(new BufferedInputStream(new FileStream("filename")));

和其他设计模式的比较

代理模式

代理模式主要意图在于对目标对象功能的控制，而装饰者模式在于对目标对象（被装饰的功能）进行扩充和组合。这个从构造方法参数可以看出，代理模式中目标对象通常是代理类直接生成，而装饰者模式通过外部使用者传入。

桥接模式

主要意图在于分离使用者和实现。

适配器模式

为解决两套不一致的接口兼容性问题。出于这个目的，代码形式上肯定有2个不一样的接口，这个和装饰者模式相差还是比较大。但和桥接模式比较容易混淆。

组合模式