这里只记录装饰器模式的**适用场合及注意事项：**

装饰者模式（Decorator Pattern），是在不必改变原类文件和使用继承的情况下，动态的扩展一个对象的功能。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。

使用装饰者模式的时候需要注意一下几点内容：

（1）装饰对象和真实对象有相同的接口。这样客户端对象就可以以和真实对象相同的方式和装饰对象交互。

（2）装饰对象包含一个真实对象的引用。

（3）装饰对象接受所有的来自客户端的请求，它把这些请求转发给真实的对象。

（4）装饰对象可以在转发这些请求以前或以后增加一些附加功能。这样就确保了在运行时，不用修改给定对象的结构就可以在外部增加附加的功能。在面向对象的设计中，通常是通过继承来实现对给定类的功能扩展。然而，装饰者模式，不需要子类可以在应用程序运行时，动态扩展功能，更加方便、灵活。

适用装饰者模式场合：

1．当我们需要为某个现有的对象，动态的增加一个新的功能或职责时，可以考虑使用装饰模式。

2．当某个对象的职责经常发生变化或者经常需要动态的增加职责，避免为了适应这样的变化，而增加继承子类扩展的方式，因为这种方式会造成子类膨胀的速度过快，难以控制。