Java开闭原则(OCP：Open/Closed Principle)

一、简介

开闭原则是面向对象设计中“可复用设计”的基石，是面向对象设计中最重要的原则之一，其他很多设计原则都是实现开闭原则的一种手段。 遵循开闭原则设计的模块有两个主要特征：

1) 对于扩展时开发的(Open for extension)，这就意味着模块的行为时可以扩展的。当应用的需求改变时，我们可以对模块进行扩展使其具有满足某些改变的新行为，也就是说我们可以改变模块的功能。

2) 对于修改时关闭的(Closed for modification)，对模块的行为进行扩展时，不必改动模块的源代码或而进制码。

二、开闭原则的思想

1) 开闭原则在Java中的体现：如何控制变化？也就是“抽象化是关键”。最为精彩的部分是由于抽象层导出一个或多个新的具体类可以改变系统的行为，因此系统的设计对扩展时开放的。对于这种抽象的方法，在Java中有接口和抽象类。

2) 实现开闭原则的关键是抽象化，在面向对象的编程语言中，可以给系统定义出一套相对比较稳定的抽象设计，此设计允许无穷无尽的行为在实现层被实现。在语言中，给出一个或多个抽象类或接口，规定出所有具体类必须提供的方法的特征作为系统设计的抽象层。这个抽象层预见了所有的可扩展性，因此，在任何扩展情况下都不会改变。这就使得系统的抽象不需要改变，从而满足了修改关闭原则。

三、关键知识点

1) 软件实体对扩展开放，对修改关闭；

2) 实现开闭原则的关键是利用接口或抽象类抽象出系统的抽象层，抽象层不变，利用实现层进行扩展；

3) 对可变性的封装，将可变的元素封装起来，防止改变扩散到整个应用；

4) 简单的说，软件系统是否有良好的接口或抽象类设计是判断软件系统是否满足开闭原则的一种重要的基准，现在多把开闭原则等同于面向接口的软件设计。