设计模式

1. 常用设计模式

- 策略模式
- 观察者模式
- 装饰者模式
- 工厂方法
- 抽象工厂模式
- 单例模式
- 命令模式
- 适配器模式
- 外观模式
- 模板方法模式
- 迭代器
- 组合模式
- 状态模式
- 代理模式
- 建造者模式

2. 设计模式的六大原则

- 单一职责原则（Single Responsibility Principle，SRP）：不要存在多于一个导致类变更的原因。通俗的说，即一个类只负责一项职责。
- 里氏替换原则（Liskov Substitution Principle，LSP）：如果对每一个类型为 T1的对象 o1，都有类型为 T2 的对象o2，使得以 T1定义的所有程序 P 在所有的对象 o1 都代换成 o2 时，程序 P 的行为没有发生变化，那么类型 T2 是类型 T1 的子类型。所有引用基类的地方必须能透明地使用其子类的对象。
- 依赖倒置原则（Dependence Inversion Principle，DIP）：高层模块不应该依赖低层模块，二者都应该依赖其抽象；抽象不应该依赖细节；细节应该依赖抽象。

- 接口隔离原则（Interface Segregation Principle，ISP）：客户端不应该依赖它不需要的接口；一个类对另一个类的依赖应该建立在最小的接口上。
- 迪米特法则（Law of Demeter，LoD）：一个对象应该对其他对象保持最少的了解。
- 开放封闭原则（Open Close Principle，OCP）：一个软件实体如类、模块和函数应该对扩展开放，对修改关闭。

3. 什么是高内聚、低耦合?

3.1. 内聚性

内聚性又称块内联系。指模块的功能强度的度量，即一个模块内部各个元素彼此结合的紧密程度的度量。若一个模块内各元素（语名之间、程序段之间）联系的越紧密，则它的内聚性就越高。

内聚性分类（低----高）：

- 偶然内聚：指一个模块内的各处理元素之间没有任何联系。
- 逻辑内聚：指模块内执行几个逻辑上相似的功能，通过参数确定该模块完成哪一个功能。
- 时间内聚：把需要同时执行的动作组合在一起形成的模块为时间内聚模块。
- 通信内聚：指模块内所有处理元素都在同一个数据结构上操作（有时称之为信息内聚），或者指各处理使用相同的输入数据或者产生相同的输出数据。
- 顺序内聚：指一个模块中各个处理元素都密切相关于同一功能且必须顺序执行，前一功能元素输出就是下一功能元素的输入。
- 功能内聚：这是最强的内聚，指模块内所有元素共同完成一个功能，缺一不可。与其他模块的耦合是最弱的。

3.2. 耦合性

耦合性也称块间联系。指软件系统结构中各模块间相互联系紧密程度的一种度量。模块之间联系越紧密，其耦合性就越强，模块的独立性则越差。模块间耦合高低取决于模块间接口的复杂性、调用的方式及传递的信息。

耦合性分类（低----高）：

- 无直接耦合:
- 数据耦合：指两个模块之间有调用关系，传递的是简单的数据值，相当于高级语言的值传递。
- 标记耦合：指两个模块之间传递的是数据结构，如高级语言中的数组名、记录名、文件名等这些名字即标记，其实传递的是这个数据结构的地址。
- 控制耦合：指一个模块调用另一个模块时，传递的是控制变量（如开关、标志等），被调模块通过该控制变量的值有选择地执行块内某一功能。
- 公共耦合：指通过一个公共数据环境相互作用的那些模块间的耦合。公共耦合的复杂程序随耦合模块的个数增加而增加。

- 内容耦合：这是最高程度的耦合，也是最差的耦合。当一个模块直接使用另一个模块的内部数据，或通过非正常入口而转入另一个模块内部。