# 设计模式六大原则（1）：单一职责原则

定义：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。

# 设计模式六大原则（2）：里氏替换原则

定义：所有引用基类的地方必须能透明地使用其子类的对象，通俗的来讲就是父类能出现的地方子类就可以出现，但是反过来就不行了。子类可以扩展父类的功能，但不能改变父类原有的功能。

# 设计模式六大原则（3）：依赖倒置原则

定义：

1）高层模块不应该依赖低层模块

2）抽象不应该依赖细节

3）细节依赖抽象

# 设计模式六大原则（4）：接口隔离原则

**定义**：

客户端不应该依赖它不需要的接口

类间的依赖关系应该建立在最小的接口上；

根据接口隔离原则拆分接口时，首先必须满足单一职责原则；

接口要高内聚，高内聚就是提高接口、类、模块的处理能力，减少对外交互。

# 设计模式六大原则（5）：迪米特法则

**定义：**

**一个对象应该对其他对象有最少的了解，一个类只需要知道自己需要耦合或者调用类的public方法即可。**

# 设计模式六大原则（6）：开闭原则

定义：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。

单例模式：

特点

1. 在整个应用程序中，一个类只有一个实例变量
2. 这个实例变量，只能在本类中创建======》私有化构造
3. 创建一个静态的方法（对外访问的接口）==》 方法其他类调用

步骤

* 1. 创建本类的静态变量

Public class A{

Static A a;

}

* 1. 私有化构造

Private A{}

* 1. 创建一个对外访问的接口

实现单例的三种方式;

0.1懒汉

0.2饿汉

0.3 双抽校验锁