## 资料地址

<https://www.cnblogs.com/lspz/p/6859991.html>

https://www.cnblogs.com/maowang1991/archive/2013/04/15/3023236.html

## 7大原则

### 单一职责

一个类一个职责

### 开闭原则

一个软件实体如类、模块和函数对扩展开放，对修改关闭。

### 里氏替换原则

继承和派生类（尽量不重写父类已实现的方法，可以用接口等其他方法绕过）

1. 子类可以实现父类的抽象方法，但不能覆盖父类的非抽象方法
2. 子类中可以增加自己特有的方法
3. 当子类的方法重载父类的方法时，方法的前置条件（即方法的形参）要比父类方法的输入参数更宽松。
4. 当子类的方法实现父类的抽象方法时，方法的后置条件（即方法的返回值）要比父类更严格。

### 依赖倒置原则

以开闭原则为基础，对接口编程，不要针对实现编程。

高层模块不依赖于底层模块，二者都应依赖其抽象细节；细节应该依赖于抽象不依赖于具体。

### 接口隔离原则

建立单一接口，不要建立庞大臃肿的接口，尽量细化接口，接口中的方法尽量少。降低类之间的耦合度，降低依赖。便于维护和升级。

### 迪米特法则（最少知道原则）

一个类应尽量少的与其他类之间发生相互作用，使得系统功能模块相对独立。就是一个类对自己依赖的类知道的越少越好。也就是说，对于被依赖的类来说，无论逻辑多么复杂，都尽量地将逻辑封装在内部，对外除了提供public方法，不对外泄露任何信息。

### 合成（组合/聚合）复用原则

尽量使用合成/聚合的方式，而不是使用继承的关系来达到复用的原则。

## 24种设计模式

### 创建者模式

#### 单例模式（Singleton）

确保一个类只有一个实例，并提供全局访问点。

使用synchronized关键字锁定对象，一定要恰当使用。

使用类的静态方法实现单例模式效果相同，但两者有区别。

#### 工厂模式（Factory）

定义一个创建对象的接口，由子类决定要实例化的类是哪一个。工厂方法让类实例化推迟到子类。

普通工厂：建立一个工厂类，对实现了同一接口的一些类进行实例的创建。

多工厂：对普通工厂方法模式进行改进，在普通工厂方法模式中，如果传递的字符串出错，则不能正确创建对象，而多个工厂方法模式是提供多个工厂方法，分别创建对象。

静态工厂：将上面的多个工厂方法模式里的方法置为静态的，不需要创建实例，直接调用即可。

#### 抽象工厂模式（Abstract Factory）

提供一个接口，用于创建相关或依赖对象的家族，而不需要指定具体类。

创建者模式（Builder）

将各种产品集中起进行管理，用来创建复合对象（某个类具有不同的属性）

即使用创建者模式封装一个产品的构造过程，并允许按步骤构造。将一个复杂对象的构建与它表示分离，使得同样的构建过程可以创建不同的表示。

#### 原型模式（Prototype）

当创建给定类的实例过程很复杂时，就使用原形模式。

与工厂模式无关，将一个对象作为原型对其进行复制、克隆，产生一个和原对象类似的新对象。

#### 多例模式（Multitude）

在一个解决方案中结合两个或多个模式，以解决一般或重复发生的问题

### 结构模式

适配器模式

装饰模式

代理模式

外观模式

桥接模式

组合模式

享元模式