01 单一职责原则、开放-封闭原则、依赖倒置原则、里式替换原则、迪米特法则

C-3 创建: 张林伟, 最后修改: 张林伟 2018-10-09 23:28

单一职责原则（SRP），就一个类而言，应该仅有一个引起它变化的原因。

如果一个类承担的职责过多，就等于把这些职责耦合在一起，一个职责的变化可能会削弱或者抑制这个类完成其他职责的能力。这种耦合会导致脆弱的设计，当变化发生时，设计会遭受到意想不到的破坏。

开放-封闭原则，是说软件实体（类、模块、函数等待）应该可以扩展，但是不可修改。（Open for extension, closed for modification.）

无论模块多么地"封闭"，都会存在一些无法对之封闭的变化。既然不可能完全封闭，设计人员必须对于他设计的模块应该对哪种变化封闭做出选择。他必须先猜测出最有可能发生的变化种类，然后构造抽象来隔离那些变化。

在我们最初编写代码时，假设变化不会发生。当变化发生时，我们就创建抽象来隔离以后发生的同类变化。

面对需求，对程序的改动是通过增加新代码进行的，而不是更改现有的代码。

依赖倒置原则：

1.高层模块不应该依赖底层模块。两个都应该依赖抽象。

2.抽象不应该依赖细节。细节应该依赖抽象。

针对接口编程，不要对实现编程。

举例：访问数据库逻辑写到一个函数（底层模块）用以复用，业务层（高层模块）进行调用。如果需要更换不同数据库或者不同连接池等等，但此时高层模块与底层模块耦合在一起。办法就是在高层模块和底层模块之间做一层抽象（接口或者抽象类）。

里式替换原则：子类型必须能够替换掉它们的父类型。

也就是说，在软件里面，把父类都替换成它的子类，程序的行为没有变化。

只有当子类可以替换掉父类，软件单位的功能不受到影响时，父类才能真正被复用，而子类也能够在父类的基础上增加新的行为。

迪米特法则

如果两个类不必彼此直接通信，那么这两个类就不应当发生直接的相互作用。如果其中一个类需要调用另一个类的某一个方法的话，可以通过第三者转发这个调用。

根本思想是强调了类之间的松耦合。