OO的五大原则是指SRP、OCP、LSP、DIP、ISP

(2013-09-19 17:36:06) OO的五大原则是指SRP、OCP、LSP、DIP、ISP。
1. SRP（Single Responsibility Principle 单一职责原则）
单一职责很容易理解，也很容易实现。所谓单一职责，就是一个设计元素只做一件事。什么是“只做一件事”？简单说就是少管闲事。现实中就是如此，如果要你专心做一件事情，任何人都有信心可以做得很出色。

2. OCP 开闭原则，很简单，一句话：“Closed for Modification; Open for Extension”——“对变更关闭；对扩展开放”。开闭原则其实没什么好讲的，我将其归结为一个高层次的设计总则。OCP的动机很简单：软件是变化的。不论是优质的设计还是低劣的设计都无法回避这一问题。OCP说明了软件设计应该尽可能地使架构稳定而又容易满足不同的需求。
为什么要OCP？答案也很简单——重用。

3. LSP（Liskov Substitution Priciple） 里氏替换原则

Liskov于1987年提出了一个关于继承的原则“Inheritance should ensure that any property proved about supertype objects also holds for subtype objects.”——“继承必须确保超类所拥有的性质在子类中仍然成立。”也就是说，当一个子类的实例应该能够替换任何其超类的实例时，它们之间才具有is-A关系。该原则称为Liskov Substitution Principle——里氏替换原则。

我们来研究一下LSP的实质。学习OO的时候，我们知道，一个对象是一组状态和一系列行为的组合体。状态是对象的内在特性，行为是对象的外在特性。LSP所表述的就是在同一个继承体系中的对象应该有共同的行为特征。
LSP讲的是基类和子类的关系。只有当这种关系存在时，里氏代换关系才存在。如果两个具体的类A，B之间的关系违反了LSP的设计，(假设是从B到A的继承关系)那么根据具体的情况可以在下面的两种重构方案中选择一种。 创建一个新的抽象类C，作为两个具体类的超类，将A，B的共同行为移动到C中来解决问题。 从B到A的继承关系改为委派关系。
4.DIP 依赖倒置原则
依赖倒置（Dependence Inversion Principle）原则讲的是：要依赖于抽象，不要依赖于具体。
简单的说，依赖倒置原则要求客户端依赖于抽象耦合。原则表述：
抽象不应当依赖于细节；细节应当依赖于抽象；
要针对接口编程，不针对实现编程。

5.ISP （Interface Segregation Principle）接口隔离原则
使用多个专门的接口比使用单一的总接口要好。广义的接口：一个接口相当于剧本中的一种角色，而此角色在一个舞台上由哪一个演员来演则相当于接口的实现。因此一个接口应当简单的代表一个角色，而不是一个角色。，如果系统设计多个角色的话，则应当每一个角色都由一个特定的接口代表。狭义的接口（Interface）:接口隔离原则讲的就是同一个角色提供宽、窄不同的接口，以对付不同的客户端。

Links