# 面向对象设计原则

**单一职责原则：**

就一个类而言，应该仅有一个引起它变化的原因。

**开放-封闭原则：**

软件实体（类、模块、函数等等）应该可以扩展，但是不可以修改，即对扩展开放（Open for extension）,对修改关闭(Closed for modification);

**依赖倒转原则：**

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。

**里氏代换原则：**

子类型必须能够替换掉他们的父类型。即：一个软件实体如果使用的是一个父类的话，那么一定适用其子类，而且它察觉不出父类对象和子类对象的区别。也就是说，在软件里面，把父类都替换成他的子类，程序的行为没有变化。

**迪米特法则：**

如果两个类不必彼此件直接通信，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个方法，可以通过第三者转发这个调用。

**合成/聚合复用原则：**

尽量使用合成/聚合，尽量不要使用类继承。

合成（Composition）和聚合（Aggregation）都是关联的特殊种类。聚合表示一种弱的拥有管关系，体现的是A对象可以包含B对象，但B对象不是A对象的一部分；合成则是一种强的拥有关系，体现严格的部分和整体的关系，部分和整体的生命周期一样。

**使用此原则的好处：**

**优先使用对象的合成/聚合有助于你保持每个类被封装，并集中在单个任务上。这样类和类继承层次会保持比较小的规模，并且不太可能增长为不可控制的庞然大物。**