## OO中的设计原则

SOLID原则：

S 单一职责原则（Single Responsibility Principle）

O 开放关闭原则（Open Closed Principle）

L Liskov替换原则（Liskov Substitution Principle）

I 接口分离原则（Interface Separation Principle）

D 依赖倒置原则（Dependency Inversion Principle）

通过应用这些原则，能够提高代码适应变化的能力。要在正确的时机和场合使用，切勿过度使用。

### [单一职责原则](http://www.objectmentor.com/resources/articles/srp.pdf)

方法、类和模块应该只具有一种职责，当需要承担其它职责时，应该把这些职责委托给其它的单职责类。



### [开放关闭原则](http://www.objectmentor.com/resources/articles/ocp.pdf)

对扩展开发：模块的行为是可扩展的。当需求改变时，我们可以对其模块进行扩展，使它能能够满足新需求变更的新行为。

对修改关闭：在对模块行为进行扩展时，不用改动该模块其它的代码。

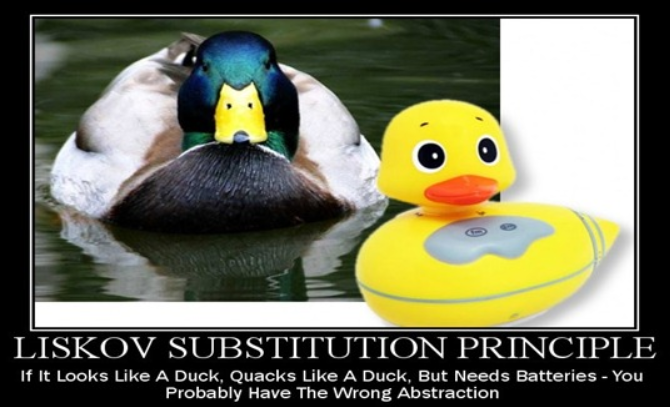
扩展点：提供应对需求改变的入口。

防止变异：识别可预见的变化点并围绕它们创建一个稳定的接口。



### Liskov替换原则

指导创建继承层次的结构，是对SRP和OCP的辅助增强。



### 接口分离原则

### 依赖倒置原则

## OO中的设计模式

## 常见的分层架构

### 分层目的

1. 高类聚，低耦合
2. 易复用
3. 可扩展
4. 易维护

### 三层架构

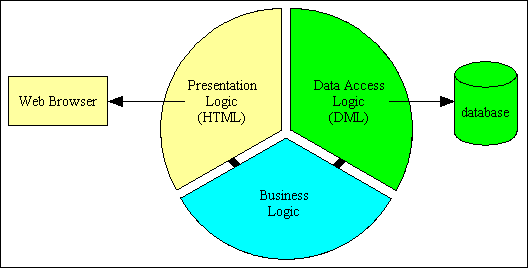
DAL: 数据访问层，主要与数据库操作相关。

IDAL：数据访问接口层，主要用于抽象数据库操作。

BLL：业务逻辑层，主要用于具体业务功能的实现。

IBLL：业务逻辑接口层，主要用于抽象业务功能。

UI: 表示层， 负责输入和输出。



### MVC

1. Model + View + Controller
2. 用户操作->View（负责接收用户的输入操作）->Controller（业务逻辑处理）->Model（数据持久化）->View（将结果反馈给View）
3. Backbone.js、SSH、Asp.net MVC

Controller

Model

View

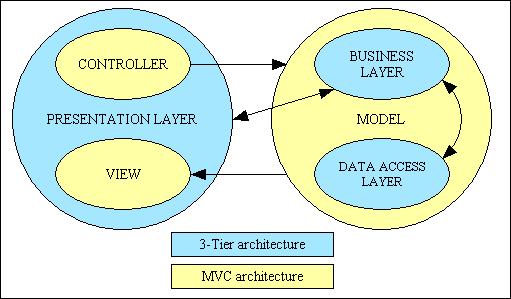
* 1. 正向MVC

Controller

Model

View

* 1. 反向MVC



* 1. 三层架构和MVC的结合

### MVP

1. Model + View + Presenter
2. MVP是把MVC中的Controller换成了Presenter（呈现），目的就是为了完全切断View跟Model之间的联系，由Presenter充当桥梁，做到View-Model之间通信的完全隔离
3. 我们熟知的ASP.NET webform、winform基于事件驱动的开发技术就是使用的MVP模式。控件组成的页面充当View，实体数据库操作充当Model，而View和Model之间的控件数据绑定操作则属于Presenter。控件事件的处理可以通过自定义的IView接口实现，而View和IView都将对Presenter负责

Presenter

Model

View

### MVVM

1. Model + View + ViewModel
2. 如果说MVP是对MVC的进一步改进，那么MVVM则是思想的完全变革。它是将“数据模型数据双向绑定”的思想作为核心，因此在View和Model之间没有联系，通过ViewModel进行交互，而且Model和ViewModel之间的交互是双向的，因此视图的数据的变化会同时修改数据源，而数据源数据的变化也会立即反应到View上
3. WPF、 VueJS、Knockout、AngularJS

ViewModel

Model

View

## 引用

1. <https://www.tomdalling.com/blog/software-design/solid-class-design-the-single-responsibility-principle/>
2. <https://www.tomdalling.com/blog/software-design/solid-class-design-the-interface-segregation-principle/>
3. http://www.tonymarston.net/php-mysql/3-tier-architecture.html