# Android 架构 MVC MVP MVVM

MVC，MVP和MVVM是软件比较常用的三种软件架构，这三种架构的目的都是分离，避免将过多的逻辑全部堆积在一个类中。在Android中，Activity中既有UI的相关处理逻辑，又有数据获取逻辑，从而导致Activity逻辑复杂不单一难以维护。

为了一个应用可以更好的维护和扩展，我们需要很好的区分相关层级，要不然以后将数据获取方式从数据库变为网络获取时，我们需要去修改整个Activity。架构使得View和数据相互独立，我们把应用分成三个不同层级，这样我们就能够单独测试相关层级，使用架构能够把大多数逻辑从Activity中移除，方便进行单元测试。

# MVC是什么

MVC是模型(Model)－视图(View)－控制器(Controller)的缩写，用一种业务逻辑、数据、界面显示分离的方法组织代码。其实Android Studio创建一个项目的模式就是一个简化的mvc模式。

## Android中的MVC含义

Model：实体类(数据的获取、存储、数据状态变化)。

View：布局文件

Controller：Activity(处理数据、业务和UI)。

## 工作原理

View接受用户的交互请求。

View将请求转交给Controller。

Controller操作Model进行数据更新。

数据更新之后，Model通知View数据变化。

View显示更新之后的数据。

## MVC的缺点

随着界面及其逻辑的复杂度不断提升，Activity类的职责不断增加，以致变得庞大臃肿。为了解决MVC的缺点，MVP 框架被提出来。

# MVP是什么

MVP是MVC架构的一个演化版，全称是Model-View-Presenter。将MVC中的V和C结合生成MVP中的V，引入新的伙伴Presenter。

## Android中的MVP含义

Model：实体类(数据的获取、存储、数据状态变化)。

View：布局文件+Activity。

Presenter：中介，负责完成View与Model间的交互和业务逻辑。

## 工作原理

View 接收用户交互请求

View 将请求转交给 Presenter(V调用P接口)

Presenter 操作Model进行数据更新(P调用M接口)

Model 通知Presenter数据发生变化(M调用P接口)

Presenter 更新View数据(P执行接口,V相应回调)

## MVP的优点

复杂的逻辑处理放在Presenter进行处理，减少了Activity的臃肿。

解耦。Model层与View层完全分离，修改V层不会影响M层，降低了耦合性。

可以将一个Presenter用于多个视图，而不需要改变Presenter的逻辑。

Presenter层与View层的交互是通过接口来进行的，便于单元测试。

## MVP的缺点

维护困难。Presenter中除了业务逻辑以外，还有大量的View->Model，Model->View的手动同步逻辑，造成Presenter比较笨重，维护起来会比较困难。

# MVVM是什么

是 Model-View-ViewModel 的简写。MVVM与MVP的结构还是很相似的，就是将Presenter升级为ViewModel。在MVVM中，View层和Model层进行了双向绑定(即Data Binding)，所以Model数据的更改会表现在View上，反之亦然。ViewModel就是用来根据具体情况处理View或Model的变化。

## Android中的MVVM含义

Model：实体类(数据的获取、存储、数据状态变化)。

View：布局文件+Activity。

ViewModel： 关联层，将Model和View进行绑定，Model或View更改时，实时刷新对方。

## 工作原理

View 接收用户交互请求

View 将请求转交给ViewModel

ViewModel 操作Model数据更新

Model 更新完数据，通知ViewModel数据发生变化

ViewModel 更新View数据

View/Model的变动，只要改其中一方，另一方都能够及时更新到

## MVVM的优点

1.提高可维护性。Data Binding可以实现双向的交互，使得视图和控制层之间的耦合程度进一步降低，分离更为彻底，同时减轻了Activity的压力。

2.简化测试。因为同步逻辑是交由Binder做的，View跟着Model同时变更，所以只需要保证Model的正确性，View就正确。大大减少了对View同步更新的测试。

3.ViewModle易于单元测试。

## MVVM的缺点

1.对于简单的项目，使用MVVM有点大材小用。

2.对于过大的项目，数据绑定会导致内存开销大，影响性能。

3.ViewModel和View的绑定，使页面异常追踪变得不方便。有可能是View出错，也有可能是ViewModel的业务逻辑有问题，也有可能是Model的数据出错。

# MVP和MVC的最大区别

在MVP中View并不直接使用Model，它们之间的通信是通过Presenter 来进行的，所有的交互都发生在Presenter内部，而在MVC中View直接从Model中读取数据而不是通过 Controller。

# MVP封装

# 复杂应用mvp封装（Activity）

# 复杂应用mvp封装（Fragment）

式样

微软雅黑 Light 小四

段落 固定值 23磅

# 目录1

## 目录2

### 目录3

#### 目录4

##### 目录5