Android 架构篇

工程师 李洪江

一 什么是组件化开发

组件化开发，也叫 AAR 开发。

组件化开发其中难点是业务组件化。

二 为什么要组件化？

Android 架构从最早的按功能分包，到后来的按业务模块分包，单一工程架构。现在主流是组件化开发和插件化开发。随着项目功能，业务越来越多，项目耦合度增强。人员更替，项目维护、学习成本高。每改一个功能，app 回归测试重复性增大。改一小部分，导致整个项目重构，花费时间太多，效率低下。

组件化优势：

1 新人学习项目成本低，代码耦合度下降，方便功能拆卸。

2 加快开发和编译效率。

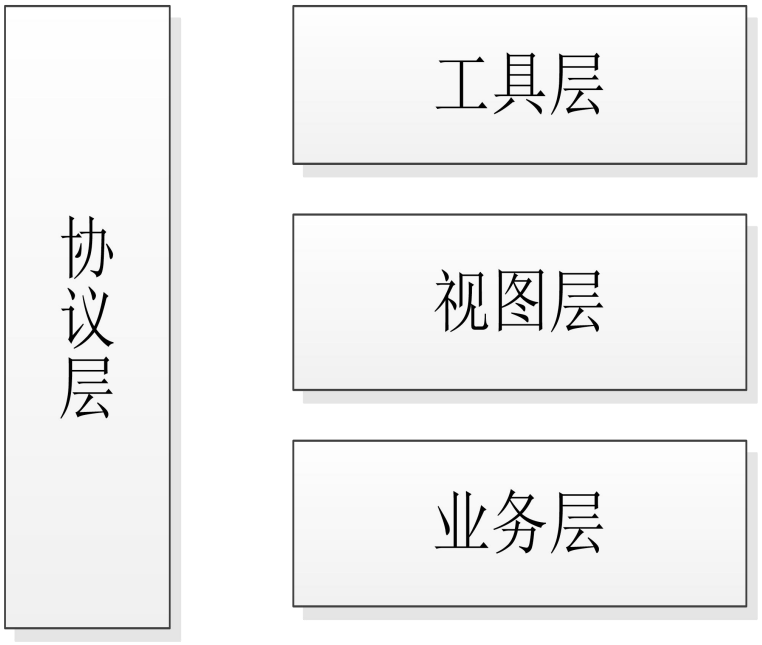
3 便于每个模块版本管理。

4 便于开发人员为每个模块写单元测试。

5 只是修改某个业务模块功能，测试人员不需要回归测试其他功能。

三 云宠 APP 组件化实践

云宠 APP 四层架构：


协议层
工具层
视图层
业务层

每层含下列库：


协议层
业务接口库
数据库
存储库
基础库
视图层
三方库
扩展库
基础库
业务层
通用业务库
具体业务库
URI处理库
WebView库
用户登录库
相册库
App主框架库
。 。 。
业务整合库
工具层
三方库
基础库

项目依赖：

```
//协议层
include ':model:coredata'          //数据协议和数据存储
include ':api:NetApi'              //网络协议接口
include ':api:RouterApi'           //业务导航接口
//业务层
include ':common:base'             //app主题包，UI设计规范定义
                                   //常用对话框，公共配置，公共UI
include ':module:url'              //app uri 映射统跳业务
include ':module:web'              //WebView处理业务
include ':module:tabnav'           //主框架导航
include ':module:user'             //用户相关
include ':module:pet'              //宠物相关
include ':module:order'            //订单相关
include ':module:family'           //家庭相关
include ':module:comment'          //评论相关
include ':module:amap'             //公共地图界面业务
include ':module:camera'           //公共图片选择业务
include ':app'                     //APP壳
//工具层
include ':plugin:PluginApi'        //环信插件，高德地图插件
include ':plugin:easeui'           //环信基础UI库
include ':plugin:shareSDK'         //QQ,微信，微博三方授权库
include ':plugin:funplugin'        //小能插件，渠道插件，友盟插件

include ':tools:PushSDK311'        //友盟二次包装库
include ':tools:XiaonengChatUI'    //小能二次包装库
include ':tools:alipay'            //阿里支付二次包装库
include ':tools:grantlib'          //6.0权限管理库
include ':tools:walle'             //美团打包渠道库
//基础库
include ':base:baserouter'         //基础UI导航库
include ':base:basenet'            //基础协议库
include ':base:baseutils'          //基础工具类
include ':base:basedata'           //基础数据库
//视图层
include ':ui:commonui'             //基础UI库
include ':ui:uiwidget'             //扩展UI库
include ':ui:thirdui'              //三方UI库
include ':ui:Emojicon'             //三方emoji库
include ':ui:ptr-lib'              //下拉刷新
include ':ui:themelib'             //沉浸式主题包
include ':ui:photolib'             //自定义相册
```

设计原则：

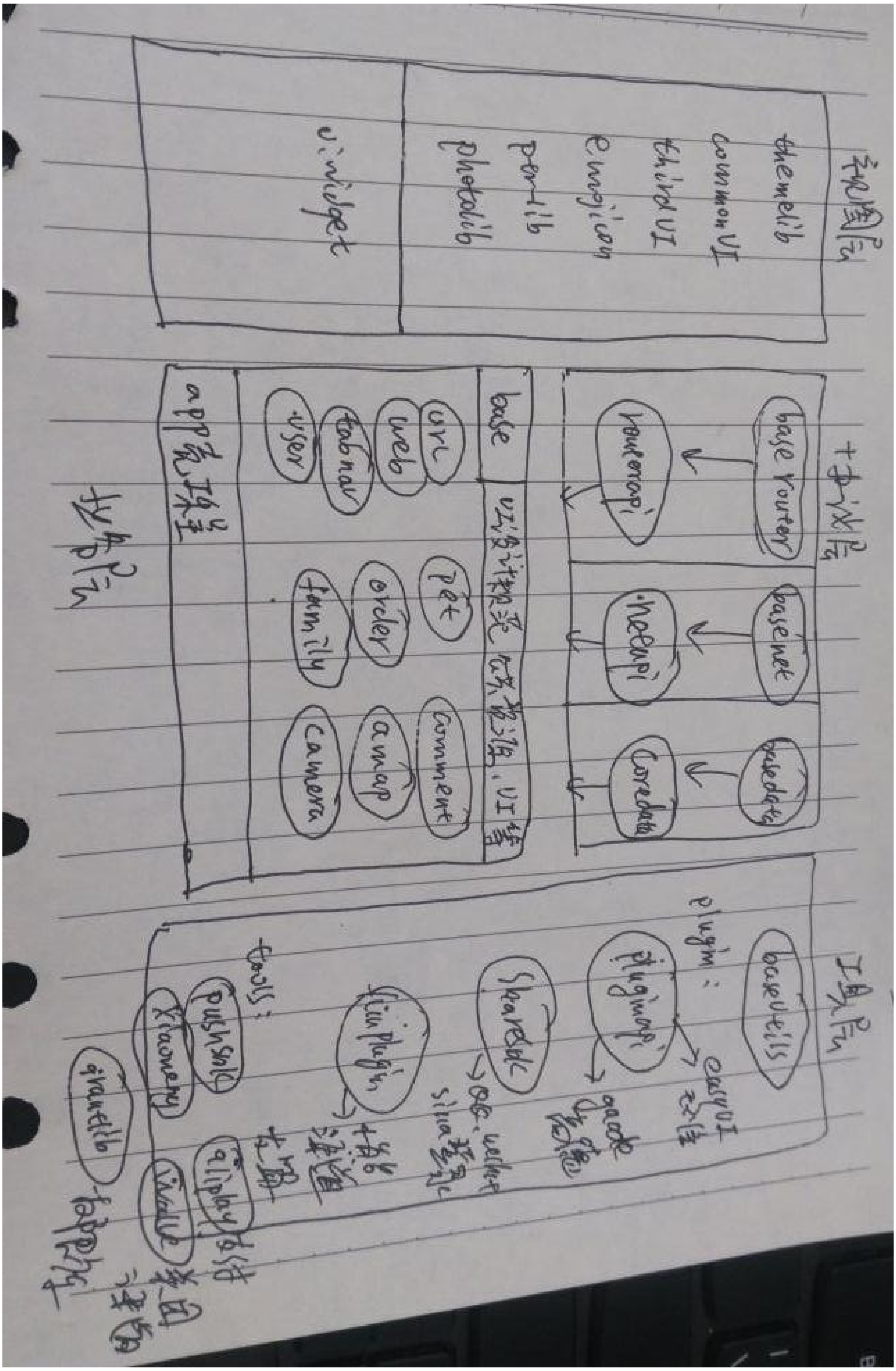
1 协议层尽量不要和 UI 层有关系。

2 业务层是通过协议层通信的。

3 业务层之间不能相互依赖。

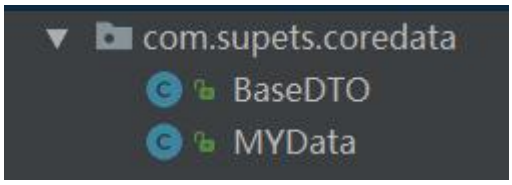
4 工具服务于其他层。

依赖关系图：


themelib
commonUI
thirdUI
portlib
photolib
uiwidget
base Router
routerapi
basenet
netapi
basedata
coredata
base
url
web
tabnav
user
pet
order
family
comment
amap
camera
baseUtils
plugin:
pluginapi
sharesdk
tools:
pushsdk
alipay
gradlelib

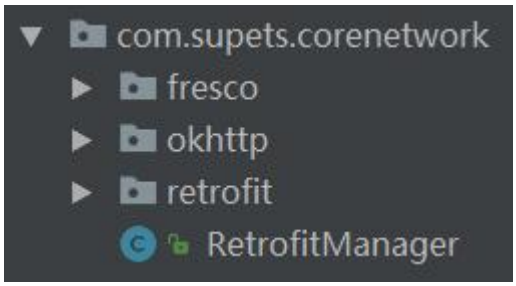
四 基础库设计

1 基础数据库 basedata



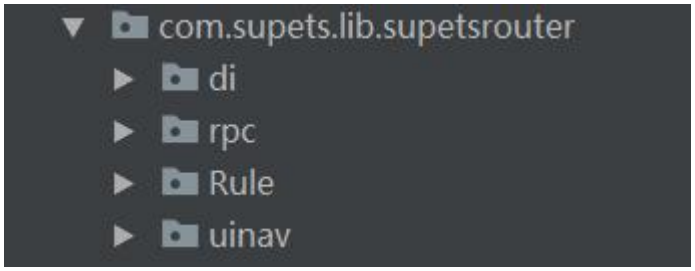
定义数据传输协议：序列化

2 基础网络库 basenet



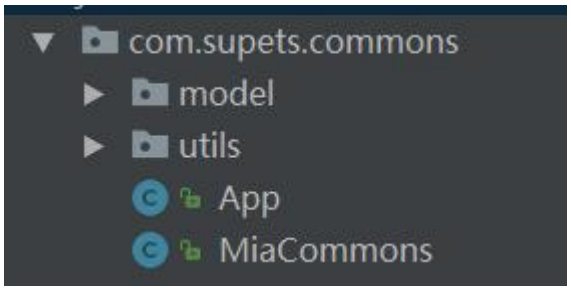
定义基础网络架构：Rest 风格

3 基础通信库 baserouter



定义了业务 UI 通信模式：
1 静态注册，使用原生 Intent 方式。
2 动态注册接口，采用 RPC 方式。
3 隐式 Intent 启动，采用 rest 风格注解框架

4 基础工具库 baseutils



主要作用：
1 App 上下文管理
2 App 协议解析工具
3 时间格式化工具
4 系统相关工具
5 UI 相关工具

五 技术使用

1 网络技术

Volley 官方网络通信库。
底层技术 SDK<9 采用 apache httpClient。SDK>=9 采用了 HttpConnection 实现。
在 SDK23 已经阉割，httpclient 实现库被独立出来，需要使用的话单独使用。
原理参考:
https://github.com/android-cn/android-open-project-analysis

Volley 的特点：特别适合数据量小，通信频繁的网络操作。
Volley 大多基于接口设计，可配置性强。常见支持 https，图片缓存实现，zip 解压缩，数据加密等使用。

Retrofit Square 出品的，是一个采用适配器架构和注解配置的 RESTful 风格设计网络库。
底层是依赖 OkHttp 实现的。
简洁好用，搭载了拦截器，方便实现日志输出，统一请求参数配合等。

2 缓存技术

GreenDao，Ormlite，RealIm

文件存储:适合一些配置类的信息，比如版本升级，渠道信息。
数据库存储：适合一些数据量大的，比如收货地址信息。
内存存储：适合一些频繁操作的数据。比如登录用户信息。

3 数据池技术
在数据改变，影响其他地方数据一致性情况下，又不需要永久存储的情景下。数据同步缓存到一个池子，再从池子拿出来。

比如，社交应用的赞，评论数，评论等同步更新。

4 运行时注解和编译时注解技术

编译时注解：butterknife，dragger
运行时注解：retrofit，EventBus，自定义 URI 框架

5 AOP 技术运用

常见写法：
1 经典的基于代理的 AOP
2 @AspectJ 注解驱动的切面
3 纯 POJO 切面
4 注入式 AspectJ 切面

比如，统计方法执行时间，android behavior 使用，一些操作需要登录再能继续。

6 注解使用
Java

```
@Override @Deprecated @SuppressWarnings
```

Android

Nullness 注解、资源注解、线程注解、值约束注解、权限注解、返回值注解、CallSuper 注解、Typedef 注解、代码可访问性注解等。

自带注解很重要。
比如:方法过时，资源标识类型，限定输入范围等。

7 泛型的使用

在封装类，方法等时候。
常见的 findViewById 方法的封装。这个在 SDK 26 已经不需要，里面已经泛型化。
在封装 baseAdapter 时候。

8 代理的使用

在 android 方面，事件的回调可以叫代理。
静态代理使用。常见写法。
动态代理使用。

9 反射的使用
反射是很多框架基础。
最早 EventBus 实现是基于反射。

运用：反射同步数据，反射获取类名。

10 图片加载
Glide，Volley，Picasso，Fresco

附录 1：常用开发工具助手

1 fir 插件
2 nexus 仓库
3 jenkins 集成
4 markdown 插件
5 git 插件
6 alibaba 开发规范插件
7 android layout ID Converter 插件
8 形状代码生成器 http://shapes.softartstudio.com/

附录 2：android studio 快捷键

大小写转换快捷键 ctrl+shift+U
提取全局变量提取全局变量:Ctrl+Alt+Full
提取全局变量提取局部变量：Ctrl+Alt+Val
提取全局变量提取方法：Shit+Alt+Method

复制块：Ctrl+D
删除：Ctrl+X
修改名称: Ctrl+Shift+R
上下移动代码 : Alt + Shift + Up/Down

注释代码(//) Ctrl + /
注释代码(/**/) Ctrl + Shift + /
格式化代码 Ctrl + Alt + L
清除无效包引用 Alt + Ctrl + O
查找 Ctrl + F
查找+替换 Ctrl + R
删除行 Ctrl + Y
扩大缩小选中范围 Ctrl + W/Ctrl + Shift + W
快捷生成结构体 Ctrl + Alt + T
快捷覆写方法 Ctrl + O
快捷定位到行首/尾 Ctrl + Left/Right
折叠展开代码块 Ctrl + Plus/Minus
折叠展开全部代码块 Ctrl + Shift + Plus,Minus
文件方法结构 Ctrl + F12
查找调用的位置 Ctrl + Alt + H
大小写转换 Ctrl + Shift + U