

# App架构设计经验谈:数据层的设计

2016-03-18 安卓应用频道

(点击上方公众号,可快速关注)

作者: Keegan小钢

网址: <http://keeganlee.me/post/architecture/20160120>

## 系列文章:

App架构设计经验谈:接口的设计

App架构设计经验谈:技术选型

一个App,从根本上来说,就是对数据的处理,包括数据从哪里来、数据如何组织、数据怎么展示,从职责上划分就是:数据管理、数据加工、数据展示。相对应的也就有了三层架构:数据层、业务层、展示层。本文就先讲讲数据层的设计。

数据层,是三层架构中的最底层,负责数据的管理。它主要的任务就是:

1. 调用网络API,获取数据;
2. 将数据缓存到本地;
3. 将数据交付给上一层。

根据这三个任务,数据层可以再拆分为三层:网络层、本地数据层、交付层。

## 网络层

网络层主要就是对网络API的封装。关于API的设计,该系列的第一篇文章接口的设计已经讲过一些。关于如何封装,可以参考Android项目重构之路系列的架构篇和实现篇,其中接口层和本文的网络层是一样的。

还有一些在前面的文章中没有提及到的,在此做一些补充。

首先是不同网络状态的处理。当网络不可用时,则不应该再去调用API;当网络可用,但不是WIFI时,有些比较耗流量的操作也应该禁止,比如上传和下载大文件;当网络状态不同时,还可以采用不同的网络策略,比如,当网络为WIFI时,当前API可以返回更多更全面的数据,还可以预先加载相关联的其他API。

其次，为了节省流量，接口的设计上可以对数据进行简化。例如，对于一些列表类的接口，可以这么设计：只返回更新的部分，比如，上一次请求返回了10条按时间排序的数据，第一条数据为最新的，id为101，当发起下一次请求时，将101的id作为参数调用API，API查到该id，发现该id之后又新增了两条数据，API则只返回新增的这两条数据。

另外，为了保证程序的健壮性，调用API时，对入参的合法性检查也是很有必要的。而且，也应该定义好本地的错误码和错误信息，保证每个错误都能正常解析。

## 本地数据层

本地数据层主要就是做缓存处理，这需要设计好一套缓存策略。设计缓存策略时，有几个问题需要考虑清楚：

1. 哪些需要缓存？哪些不需要缓存？
2. 缓存在哪里？数据库？文件？还是内存？
3. 缓存时间多长？

### 哪些需要缓存？

将所有数据都缓存是不明智的，不同的数据应该有不同的缓存策略，比如一个电商App，首页的商品列表数据应该缓存，而且缓存时间应该比较长，而每个商品的详情数据就没必要缓存或缓存时间很短。对于一份数据需不需要缓存，判断标准可以是：用户查看该数据的频率高不高？首页商品列表是用户每次启动都会看到的，而每个商品的详情用户最多只看几次。

### 缓存在哪里？

从内存读取数据是最快的，但内存非常有限。因此，内存一般只用来缓存使用频率非常高的数据。

文件缓存主要就是图片、音频、视频了。

数据库可以保存大量数据，主要就是用于保存商品列表、聊天记录之类的关系型数据。

然而，不管缓存在哪里，都需要限定好缓存的容量，要定期清理，不然会越积越多。

### 缓存时间多长？

首先，每份缓存数据都应该设置一个缓存的有效时间，有效期的起始时间以最后一次被调用的时间为准，当该数据长时间没有再被调用到时，就应该从缓存中清理掉。

缓存的有效时间应该设多长呢？可以短至一分钟，长至一星期甚至一个月，具体因数据而异。一般内存的缓存时间不宜太长，程序退出基本就要全部清理了。文件缓存可以设置保留一天或一个星期，可以每隔一天清理一次。数据库缓存再久一些也无所谓，但最好还是不要超过一个月。

## 交付层

交付层其实就是一个向上层开放的交互接口层，是上层向数据层获取数据的入口。上层向数据层请求数据，它是不关心数据层的数据是从缓存获取还是从网络获取的，它只关心结果，数据层能给到它想要的的结果就OK了。因此，交付层主要就是定义一堆开放的接口或协议。

如果接口或协议非常多，那么，将接口或协议按照模块划分也是有必要的。比如微信，按模块划分有：IM、公众号、朋友圈、钱包、购物、游戏等等。模块之间应该尽量相对独立、松耦合。

## 写在最后

数据层如果再扩展，还可以再加入日志管理，这里就不再展开讲了。上面内容讲得也比较乱，有哪里讲得不好的地方欢迎吐槽。

---

## 安卓应用频道

专注分享安卓应用相关内容



微信号: AndroidPD



长按识别二维码关注

---

伯乐在线 旗下微信公众号

商务合作QQ: 2302462408



微信扫一扫  
关注该公众号