**宁波理工学院**



**毕业设计（论文）**



题 目 **基于Android的健康信息服务系统**

姓 名 **卓修武**

学 号 **3120411040**

专业班级 12**信息与计算科学1班**

指导教师 **刘启玉**

学 院 **信息科学与工程学院**

完成日期 **2016年 5月10日**

# 摘 要

健康是每个人都爱关注的话题，近几年手机硬件水平和网络水平飞速发展，目前手机已经成为了人们上网获得健康资讯的主要途径。而在手机系统市场上Andorid系统的所占比例最大，因此本文将基于Android系统设计开发一款健康资讯类App。

**关键词：**可拓方法；物元模型；关联度；化工；安全评价

# **Abstract**

By applying matter-element theory, extension set theory and dependent function calculation of extension method, the elassica1 and 1imited matter-elements of chemical evaluation unit were set up, and matter-element model of the chemical evaluation unit was established as well in this paper. By using the element model and selecting the appropriate factor as the evaluation, the hazard of the evaluation unit was evaluated. The hazard assessment was conducted on eight sample data of the evaluation unit, and reasonable assessment results were obtained. Compared with assessment results of other assessment methods, this method can reflect the actual hazard degree of evaluation unit more truly. It is proved that the extension theory to be applied to the safety assessment of chemical industry is feasible and effective.

**Keywords:** Extension method；Matter-element model；Degree of association；Chemical industry；Safety assessmen

# 绪论

## 研究背景

根据调查，2014年新网民最主要的上网设备是手机，截止至2014年12月，我国网民的规模达到了5.57亿，使用率为64.1%，超过以往的PC设备的上网方式[5]。时下，手机的功能日益强大，正逐步取代了相机、地图、计算机、等设备，几乎无所不能[6]。这也为健康类App的发展提供了硬件基础，如基于移动传感器的运动类App，基于热传感的体温检测，基于相机的病症远程看诊等。

在几年前，中国的移动网络速度还不够快，无法支持大流量数据的快速浏览，但如今随着4G网络的发展和宽带资费的降低，越来越多的人可以随时随地地浏览。

目前移动端全球市场上主要使用的操作系统有Android、IOS、Windowphone。Symbian系统已经基本退出江湖。 在目前实际的开发市场中，谷歌公司的Android和苹果公司的iOS正在处于二分天下的竞争形势[8]。

谷歌公司在2007年正式推出Android操作系统，该系统在短短几年内就得到了广泛应用[10]。安卓操平台的优势包括其开发性，它允许任何移动终端厂商加入到Android平台。这也是目前Android操作市场仍然占据市场占有率第一的原因。安卓的开发语言目前官方采用的是Java语言，Java语言的强大功能和极佳的平台适应性也是Android的一大优点。Android为开发者提供了巨大的软件库，这些库提供了手机功能及特点的API、网络访问功能、提供了包括功能强大的WebView在内的全套视图对象。

可以看出目前移动端的技术开发已经非常成熟，也具备了强大的硬件和快速的网络条件，在不同操作系统中，无论是iOS开发还是Android开发都已经是非常成熟。

## 研究意义

目前人们生活在压力大、节奏快的环境下，很难抽出充足的时间去关心个人的健康问题，本课题以Android系统为基础搭建一个健康资讯App，基于移动端的健康信息服务使得人们可以随时随地得获取健康信息，帮助人们节省时间，提高公众的健康意识。

基于Android的健康信息服务系统将海量的分散的健康医疗信集中起来，实现信息的共享和增值，可以让用户在统一的环境下从不同的数据来源获取需要的健康信息，对于推动健康事业的发展具有重要意义。

## 研究现状

在功能上，目前市场上健康服务类App的功能繁多。具体的功能分类如下所示:

1. 由于市面上用户量比较大的医疗健康类App的前身大多是由传统互联网领域的专业健康网站[12]，比如丁香园网，39健康网等，这些健康类App在内容上基本上和以往一样，主要的功能是健康讯息推送。此类App向用户推送一些实时的健康新闻，精辟的健康知识点，还包括常见疾病和药品的讯息等。
2. 有些App比如“好大夫在线”和“掌上120”还实现了直接通过电话的方式进行咨询，这也满足了一些用户的紧急需求。用户量比较大的“春雨医生”还开发了智能的自诊系统，自诊系统可以根据年龄，性别，症状来查询患病的概率。
3. 另外，有一些大型的专业医疗机构也开发了自己的健康传播类App，其中比较知名的包括“温附一”、“福医大附一医院医护”、“苏大附一助手”等，这些App的主要功能是与自身的服务系统绑定的，主要功能是为了预约挂号和就诊导航，它的面向群体是在本医院就诊的患者及其家属，因此用户群体较小，受众面窄，不能作为健康传播类App的主流。
4. 还有一类运用主打健身跑步方向，比如联想推出的“乐疯跑”跑步类运用，?此类运用主要提供个人锻炼的训练计划，和运动的数据记录。通过个人的运动时长、里程，计算卡路里的销毁。一些跑步类App还内置了社交功能，可以发起团队跑步的方案，邀请附近的人加入，通过按里程数据排名等方式，激发用户的荣耀感，以增加用户粘性。

在质量上，由于目前移动医疗行业法律规范监管制度不全[13]，而且目前市面上存在的绝大多数App还处于运营阶段，尚未走出中国传统互联网的盈利方式[14]，也就是靠卖广告赚钱，并未深入到法律层面。数量庞多的App鱼龙混杂，不仅让用户无所适从，还存在的信息质量难以保证的问题，大量的错讹信息的问题。在有些App中夹着这涉及色情的信息，以此来达到吸引用户的目的，这对青少年儿童的身心健康有着不良的影响。由于此类App也尚未纳入政府的监管，某些App为了盈利，在App中推广广告，谎称一些事物或者保健品能够预防某些疾病，更加危险的是，有些问题的解答来自一些普通网站的搜索，解答的内容良莠不齐，如果使用这类App对我们的健康造成损害时，无法通过法律途径进行有效的维权。虽然市面上有些医疗健康类 APP拥有了大量的专业医生资源，但是还是不允许给用户进行诊断和开处方的，只能给用户进行健康指导和就医注意事项[15] 。

在用户量及发展前景上，“春雨掌上医生”在今年三月越活跃用户就达到了260万人，但是也是在经过两年的煎熬和摸索后，才开始了商业化的步伐。虽然，相对其他领域的移动App上，此类App的用户了解与接收程度不高，根据赵曙光等人在《中国健康传播》中得出的结论，“公众主动搜索健康生活方式信息的行为频率与年龄、收入水平呈现显著的正相关关系”[16]。不过，随着公民对于健康知识的需求越来越高，相信此类App的用户市场量将逐步提升。在今年的10月21日，腾讯科技报道了女性健康类App“大姨妈”获得了海通开元、汤臣倍健及创始人柴可追加的投资，投资额达到了1.3亿人民币，其估值也达到了两亿美元。

可以看出目前移动健康类App在开发上已经具有较为成熟的路线，并且越来越受到投资者的青睐，具有广阔的发展前景。

## 论文研究方法

### **2.4.1 改善程序性能**

考虑到移动端用户的特性，使用移动流量的资费是较高的，并且基于安卓的应用程序内存有限，所以在安卓程序开发中，如果不仔细考虑到内存的缓存问题，程序经常会遇到OOM而导致奔溃。以下是程序在开发时应该注意的性能问题：

1）多级缓存机制

程序内部对于从网络上获取的向用户展示的信息类数据，采用多级缓存的机制缓存到本地和内存。使用多级缓存可以提高程序的响应速度，降低用户浏览的不必要花销。

2）图片的压缩处理

对网络获取的图片加载到内存前，先进行压缩处理，防止内存溢出，导致程序奔溃。如果在程序在运行过程中奔溃，那对于用户来说是非常不好的体验。

3）ListView的优化

在安卓程序中，对于信息展示类App，使用最多的View就是ListView，ListView做多一个ViewGroup，包含许许多多的子View，ListView在滑动的过程中需要充分考虑View的回收和复用问题。ListView性能对于整个程序的运行流畅具有很大影响，所以需要充分考虑，ListView的优化。

### **2.4.2** **程序架构的设计**

在开发系统时，目前开发人员使用较多的是基于MVC的设计模式来开发，模型-视图-控制器(Model-View-Controller)是Xerox PARC在二十世纪80年代为编程语言Smalitask-80发明的一种软件设计模式。目前这个软件设计模式已经被广泛应用于软件设计的各个方面了。MVC模式主要是把应用程序分为了三个互相联系又相对解耦的部分，即模型、视图和控制器。 MVC模式的应用主要是为了增加代码的重用性、减少数据表达、数据描述和应用操作之间的耦合性。同时，也使得软件的可扩展性、可维护性、灵活性和封装性大大增强[4]。图2.3是一个简单的MVC模式架构图。

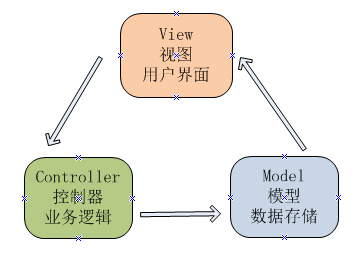


图2.3 Android的MVC模式架构图

在Android中，整个Android提供的SDK本身就是一个良好的MVC模式的体现。以下是MVC在 Android中的对应关系。

(1)模型层(Model) 对于数据库的操作、对网络的操作等都应该是放在Model层处理的。

(2)视图层(View)，一般开发人员通过编写xml进行界面的描述，xml文件在使用的使用可以非常方便的引入，而且也利于后期的修改界面的修改，逻辑中如果与界面对应的id不变的话，则代码就不需要修改，这大大增强了代码的可维护性。其实Android还提供了强大的JavaScript+html的方式作为view层，Android的SDK提供了 Java 和 JavaScript之间非常方便的通信实现。

(3)控制层，负责具体的业务逻辑调用，视图层访问模型层的数据是通过控制层来操作。

由于在Android的实际开发过程中，V层通常视图层由res目录下各个layout文件来实现，但是layoutd能够做的事情十分有限，通常还需要在视图载体，也就是Actrivity/Fragment 中编写大量的视图逻辑，这中间视图的变化往往需要通过控制层调用访问模型层的数据，数据再回传给视图层修改视图显示，这部分代码逻辑往往直接在Activity中编写，造成了Activity既有视图逻辑的代码又有业务逻辑代码的调用，使得Activity类十分臃肿，后期难以维护。对于一个拓展稳定的应用来说，我们需要定义分离各个层，主要是UI层、业务逻辑层和数据层，因为业务逻辑是经常需要变化的。于是为了使得Model和View的代码分离，目前衍生了MVP(Model-View-Presenter)的架构来解决代码耦合的问题。

以下是一个简单的MVP架构图。



MVP模式的优点在于,将业务逻辑部分从视图中剥离，View不需要也不要直接访问Model层，所有跟业务逻辑有关的操作都是在Presenter中定义的，Presenter最为View和Model的纽带将2者连接起来，他们直接的交互只有Presenter，而在实际开发中，View和Presenter之间的关联是通过接口来描述的，这样的好处是View既可以是Activity也可以是Fragment这不影响V-P直接的关系，因为他们是基于抽象的，这使得Presenter可以通过细分功能来实现复用，减少重复的代码编写。本文将在实际项目开发过程中在复杂的模块功能使用MVP的架构模式来实现解耦。

### **2.4.3** **Android数据交换格式**

Android客户端与服务端一般是采用Http的方式进行数据通信[5]，与PC浏览器访问Web服务器时采用浏览器来解析从服务端返回的数据不同的是，Android客户端是需要开发人员在程序中自行编写解析服务端返回的数据，所以在Android与服务端通信过程中，制定数据的交互格式就很有必要，这也是双方进行通信的基础。

目前网络上主要的数据传输格式有XML和JSON两种[6]，XML是默认的、预订数据格式。尽管其他的数据格式也可以使用，但是基于XML结构的可用性进行规划是一种最安全的方式。像 Android这样通常在蜂窝数据网络上工作的资源受限平台不能存储和解析大量XML数据。目前，越来越多的Internet API供应商提供 JSON 作为一种数据格式。JSON(JavaScript Object Notation)在 Ajax(Asynchronous JavaScript and XML)网页编程社区中享有盛名。JSON是一种轻量的数据交换格式，冗余度小，且编程简单[7,8,9]，它主要有一下一些特点:

1. JSON具有相当高的可读性，具有简易的语法。
2. 数据体积小，传输快，方便前台数据操作。
3. JSON非常流行，所以Android中内置了解析JSON的操作类，方便用户调用。
4. JSON数据格式在定义时舍弃了XML数据格式中的使用各级标签来区分数据的做法，这种方式大幅度的减小了数据的冗余度。

本文章的系统将采用JSON来传输数据，使用JSON传输的数据是在面向对象的基础上设计的。首先需要为系统所使用的数据设计各种符合JavaBean的对象，JavaBean是一种Java语言写成的可重用组件。由于安卓程序是采用Java语言编写的，在编写安卓端程序代码时，我们通常需要设计许多JavaBean类。在JavaBean类设计编写完成的情况下，我们根据不同业务逻辑所需要的数据，在JSON中将数据以对象的形式加入，在Android SDK中，谷歌为我们设计了一个方便的解析类，它位于org.json包下。安卓端在与服务器端的交互中，网络请求获取到JSON数据，本质上也只是一个字符串，我们通过JSONObject或者JSONArray等Java类，将之转换为JSON数据对象。在设计JSON数据时，以JavaBean的形式传输数据的好处是，我们可以将这些JSON数据读写的逻辑代码抽象成公共的库，只要是符合JavaBean的数据，我们都可调用统一的读写API。

在开源社区也存在许多非常优秀的JSON解析第三方库，比如谷歌的Gson、fastjson等，这些开源库的内部一般都利用类的反射的机制来实现。这类第三库的方法调用也非常简单。

## 论文研究内容

本文将研究如何基于Android平台设计开发一款移动健康资讯服务App，该App可以方便用户查询和了解健康方面的资讯，帮助用户养成良好的健康生活习惯。

本文的研究内容主要包括以下几个方面：

1. 明确设计该系统的意义和价值，对该系统进行了需求分析、技术调研、系统设计
2. 根据设计实现该客户端，客户端模块分为用户、健康资讯、疾病库、附近医院、健康问答，5个模块，除此之外还有常用的程序个性化设置。
3. 了解和使用MVP的设计模式，配合RxJava、Retrofit等开源库设计和实现一个易拓展、测试的架构
4. 数据采用多级缓存机制，提高App的体验效果
5. 基于Material Design设计一个良好的UI效果

# 健康信息服务系统的设计

## 开发环境搭建

### 在AndroidStudio下基于Gradle构建项目

Android官方推荐使用Gradle来构建Android项目，使用Gradle能够很好的管理项目依赖和一些配置信息。本项目的AndroidStudio版本是在2.1，Gradle版本为2.4。在Android官方网站 developer.android.com有详细的使用AndroidStudio搭建项目的基本教程。

我们直接通过Android Studio “New Project”创建一个新的工程，此时工具已经为我们创建好了.gradle、.dea\、app等目录，我们的主工程项目就是在默认的app目录下开发的，在app目录下的build.gradle脚本文件中我们需要定义项目工程的一些基本信息。首先本文的Android项目最低支持Android版本14,开发的SDK版本是23,所以对应的gradle配置文件如下：

apply **plugin**: **'com.android.application'**apply **plugin**: **'me.tatarka.retrolambda'**apply **plugin**: **'idea'**  
android {  
 compileSdkVersion 23//编译代码成字节码使用的SDK版本  
 buildToolsVersion **"23.0.2"** defaultConfig {  
 applicationId **"com.think.awhealth"** minSdkVersion 14//最低支持的Android版本  
 targetSdkVersion 23//表示经过测试可以运行的笨笨  
 versionCode 1  
 versionName **"1.0"** }  
 sourceSets {  
 main() {  
 jniLibs.srcDirs = [**'libs'**]  
 }  
 }  
 repositories {  
 flatDir {  
 dirs **'libs'** *//this way we can find the .aar file in libs folder* }  
  
}  
dependencies {//依赖库  
 compile fileTree(**include**: [**'\*.jar'**], **dir**: **'libs'**)  
 testCompile **'junit:junit:4.12'**  
compile **'com.android.support:appcompat-v7:23.0.1'** compile **'com.android.support:support-v4:23.3.0'** compile **'com.android.support:cardview-v7:23.3.0'**}

　　在Android N版本中，官方支持了Java lamda语言在Andorid环境的使用，不过在本文项目开发的时候Android N尚处于Preview版本，所以本文使用的Android　SDK23来构建项目，为了使用lamda语法来简化代码的编写，使用了支持lamda语法的第三方工具库**retrolambda'**来编写项目。

### 使用Gradle进行多模块构建

Gradle支持多模块构建项目，通过将将不同模型层的代码置于不同模块，可以更好地组织不同模型层的代码。

使用多模块构建项目的方式很简单，在本文的项目中，使用Android Studio工具，通过New Module 创建 data、和domain模块。Android Studio会自动在项目根目录的setting.gralde文件中，将data和domain模块包含进来。脚本如下。

include ':app',  
 ':data',  
 ':domain'

## **健康信息服务系统的需求分析**

本文通过对健康信息服务系统的现状以及用户需求调研，将从用户功能上进行详细的需求分析。

根据用户对常用功能的需求，系统除了提供最基本的登入注册功能外，最主要的是提供方便的疾病、药店、类的信息查询功能。考虑到用户的交流和积极性，还应开发具有交互的功能。系统的主要功能需求如下：

1）用户的注册登入

为用户提供注册、登入功能，对于登入用户和游客用户提供不同的功能。移动端的用户来说，在屏幕上输入用户密码很不方便，并且移动端的用户通常不希望为一个资讯类App进行繁琐的注册流程。所以本文系统应该将大部分的功能的权限设定为游客就可以使用，比如药品、疾病信息的查询、信息送等。

2）疾病的查询

为用户实现疾病的查询功能。用户最常见的需求是查询疾病类的相关信息。用户可以输入的信息应该是多样的，用户可以通过输入疾病的病名、疾病症状、和不适症状进行查询。系统根据用户的键入信息返回该疾病的相关信息。

3）基于用户位置，提供附近的药店医院信息  
 用户通过浏览相关医药信息后，可能希望获得附近的药店和医院信息，及时的购买药品或到医院检查，系统可以提供相关的药店和医院的具体信息和路线，为用户的出行带来方便。

4）推送健康资讯信息  
 在现代社会生活中，人们身体不健康的很大一部分原因是因为不健康的生活方式。本文的系统不仅仅提供健康的疾病和药品医院功能，还应该向用户推送介绍有关健康的生活方式，让用户了解健康生活的重要性，帮助用户养成良好的生活习惯。

5）提供用户交互的模块

用户可以发起健康方面的问题，其他注册用户可以对该问题进行讨论。发起问题的用户通过浏览回答，可以采纳最佳的回答，以此增加用户的互动，提高用户粘性。

## **健康信息服务系统的模块设计**

根据初步的系统需求分析，本文将整个系统从功能上划分为五个模块，系统功能结构如图 2.1所示。

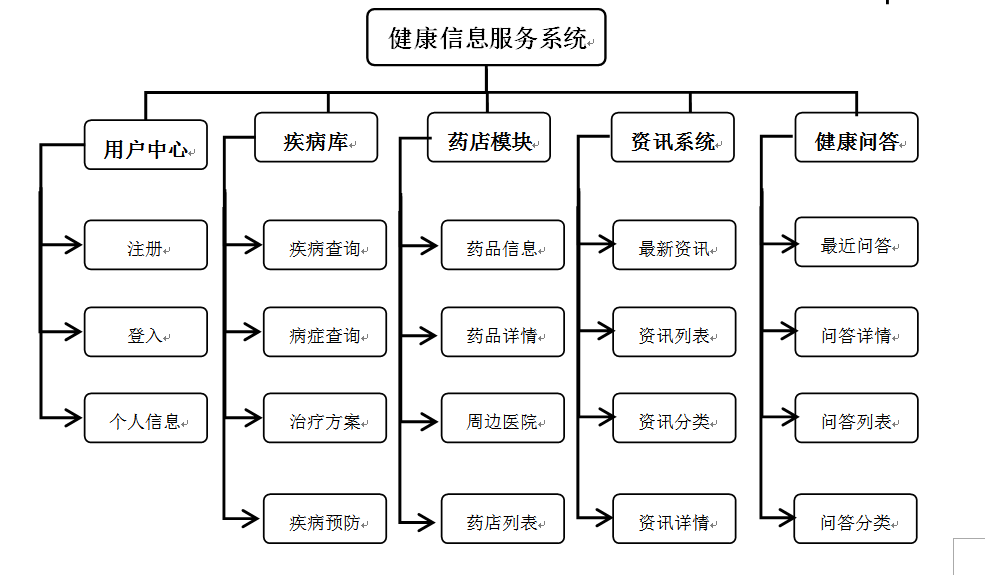


图2.1 健康服务系统功能图

系统各个模块的主要功能如下：

1）用户中心模块：提供注册、登入、注销等功能，将用户角色分为游客和登入用户，登入用户和游客都可以正常使用APP的功能，系统只对登入用户提供回复帖子，发起话题等特殊功能，其他查询类功能、推送类功能任何角色都能使用。  
 2）疾病库模块：为App用户提供方便的疾病名查询，疾病症状查询，疾病治疗方案等多种查询功能，用户可以通过输入疾病名、不适症状进行疾病的自查，服务端需要根据用户的请求内容返回疾病的治疗方案或者疾病信息等。  
 3）药店模块：当用户通过浏览资讯得到某些药品名，用户可以通过此模块进一步查询药品的价格、不良反应、禁忌等信息，基于GPS和百度地图，可以为用户提供周边的医院、药店列表和信息，  
 4）资讯系统模块：通过主动地向用户推动健康生活资讯内容，帮助用户养成良好的生活习惯，获得健康的养生方式。主动定时地发送有营养的内容也有利于提高用户的粘性。健康资讯系统具体的分类包括最新资讯、资讯列表、资讯分类、资讯详情。  
 5）健康问答模块：为App提供一个健康问答的模块，用户可以发起有关某个健康问题的问答，其他用户可以回复此问题，发起问题者可以通过采纳问题提高回答者的分数。这个功能可以增强用户与用户之间的互动，用社区互动的方式提高用户健康问答的兴趣。

6) 日志模块：在功能模块之外，将设计实现日志上传功能以便统计分析用户行为、和程序异常的信息收集。

在此基础上，本文将利用Android Studio工具，开发相应数据库，并实现用户管理，疾病、药店（医院）的信息查询，健康资讯的展示与推送、用户问答的功能。在界面设计上做到简洁大方，在开发时充分考虑程序的性能，使用优秀的第三方开源库缩短软件的开发周期。

## 系统相关技术框架

### 函数响应式编程框架RxJava

RxJava是函数响应式编程在Java平台的一个实现库。响应式编程可以加深代码的抽象程度，使得用户更专注于定义与事件相互依赖的业务逻辑，而不是把大量细节放在实现的细节上。RxJava是基于观察者模式实现的，最核心的内容就是Observable(被观察者)和Subscribe(被观察者)。Observable发出一些列的事件，Subscribe接受和处理这些事件。在RxJava的实现中，一个Observable可以发出零或多个事件，直到结束(onComplte())或者发生异常。每发出一个事件，就会调用Subscribe的onNext()方法，最后调用onComplete()或者onError()结束整个事件流。

RxJava通过良好的接口设计和强大的转化符可以帮助开发者少写很多常用的代码，使用了RxJava就不需要通过接口回调来处理事件通知的情况，这使得代码更加简洁，避免了CallBack Hell的情况。

### 简化Http请求Retrofit库

Retrofit 是著名的Square公司开源的网络请求库，基本是目前Android开发的标配，它的官方有一句简短的介绍是“A type-safe REST client for Android and Java”,Retrofit非常适合应用在服务器接口符合REST规范的情况，但是对于普通的接口使用起来也是没有区别的。Retrofit使得我们只需要定义一个Java interface就可以发起Restful API请求。

Retrofit最新的版本是2.0，2.0版本和之前的版本有很大的不同，它支持了请求接口直接返回RxJava的Observable类型，并且提供了通过构建者模式使得用户可以自己选择Json解析库和Htpp Client。基于插件化的设计思路使得用户可以根据需要搭建出适合自己的一套网络请求配置。

### OkHttp网络请求库

OKHttp是安卓6.0以后的官方Http Client，在6.0之前Http Client有2个选择，一个是Apache实现的HttpClient,另一个是Android SDK自带的HttpUrlConnection，这2个库都或多或少有一些缺点或者Bug。

HttpClient拥有众多的API，代码实现较为稳定，但是同时也是因为API层级 复杂数量较多，使得我们很难在不破坏兼容性的情况下对它进行升级和拓展，所以目前Android团队在提升和优化HttpClient方面的工作态度并不积极。在Android6.0，官方直接将该库标记为Depercated，并从Android库中移除了。

HttpUrlConnection是一种多用途、轻量级的Http客户端，使用它来进行Http操作可以适应大多数的应用程序。但是在Android2.3版本之前，HttpUrlConnection存在一个直接导致连接池复用失效的Bug，这也导致开发人员如果需要适配2.3版本以下要做更多的工作。

针对以上情况，Android官方推出了OkHtpp库，并建议开发者使用该库。OkHttp的接口设计非常简洁明了，功能也非常强大。它提供了对最新的Http/2协议版本和SPDY的支持，这可以使得对于同一个主机发出的所有请求都可以共享相同的套接字连接，提高了连接效率。OkHttp默认提供了对GZIP的支持，这可以降低传输的内容大小，降低客户端网络流量的花销。OkHttp还提供了对Http响应的缓存机制，这对于手机客户端非常有用，对于相同的Api请求，并且通常响应内容不变的情况下，可以对响应的内容进行缓存，当再次发起给Api请求时，会判断缓存中是否存在，存在就不发起Http请求了。

### Fresco

## 项目架构设计

　　在整个Android项目起步之前,我们需要粗略地定制好项目的结构，通常一个项目可以分为以下这个几个部分：

界面层:负责加载视图和呈现视图的Activity、Fragment、Widget（复杂的自定义组件）、AndroidView(系统SDK提供的基本View组件)等,在使用到列表视图时，Android提供的控件通常会使用适配器的模式来设计，所以还会有ListView的Adapter类。

视图层：不同数据对应的JavaBean类，数据层需要从不同的数据（网络或者数据库）来源获取数据并暴露接口给上层调用。

业务逻辑层:一些具体的业务逻辑操作类。

数据的交互过程基本是：当用户操作或者视图载入时，视图层调用业务逻辑层的代码，当业务逻辑层调用视图层暴露的数据访问接口，此时数据层从不同的数据来源获取数据，这中间可能会有数据的本地保存和内存缓存功能，数据层获得需要的数据后，通知业务逻辑层，这里的业务逻辑层中在通知视图层，视图层通过业务逻辑层返回的不同结果呈现出不同的视图。

通过以上的简单分析，在开发之前，创建以下这些包名来组织项目结构，在项目开发后如果有后续需要再做添加。

* entity：所有的实体类
* db 数据库的操作类
* adapter 视图适配器
* utils 一些通用的帮助类
* widgets 一些自定义View
* constant app全局使用的一些静态变量
* api 与网络接口交互的类

项目的整体架构可以由图来表示，引入的RxJava库贯穿全层，将数据理解为被订阅的对象，View和Presenter作为订阅者，作为系统的表现层，其中Presenter层负责和Domain层交互，这使得View层的代码真正做到只负责视图逻辑的部分。在业务逻辑简单的时候，Domain层也可能直接省略，Presenter直接从Respository层中获取数据。Respository的数据来源于Disk(本地)和Cloud(服务器)。

表现层(Presentation Layer)

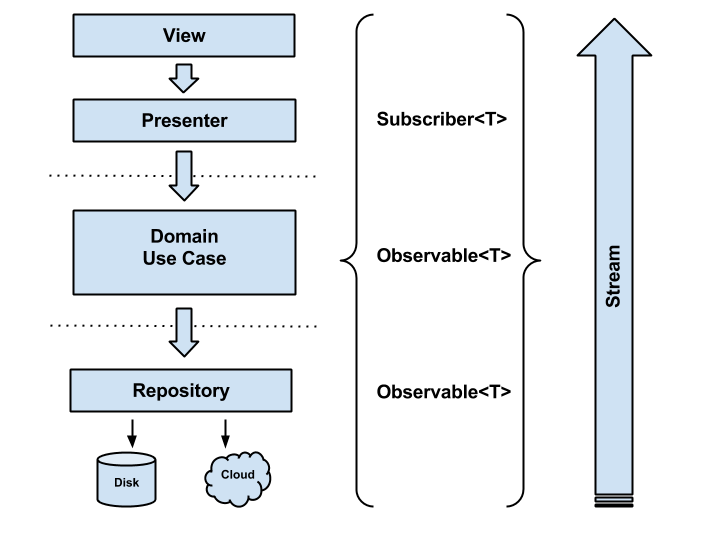
表现层，表示的是视图和动画的相关逻辑，表现层使用了MVP模式，这可以使得fragment和activity作为View其内部除了UI逻辑之外没有其他逻辑。由于Android的单线程特性的原因，不希望在UI主线程做太多工作，所以尽量将Presenter所做的工作通过 RxJava库置于专门的线程池中。

领域层(Domain Layer)

领域层是业务逻辑所处的层级，我们希望做到在该层的代码是由纯Java的代码实现的，使得它没有任何的Android依赖，这样做的原因一个是分层解耦，另一个是使得这一部分的代码因为不需要Android环境而更易于测试。

数据层(Data Layer)

一个应用最重要的内容就是需要展示的数据，应用的所有数据都来源这一层级的UserRespository实现。项目中这一层的实现采用了Respository Pattern的实现，其主要策略是通过一个工厂根据一定的条件选择不同的数据来源。比如这里的Disk和Cloud。在项目的代码实现中，会将Respository和Disk数据源、Cloud数据源实现相同的一个数据行为抽象接口。



## Retrofit 简化Http请求

项目使用了Retrofit简化Http代码的编写，传统的Http请求，需要在代码中调用方法指定Http的请求方式，如Get、Post等，如果参数复杂，还需要更多代码指定参数来完场整个Http表单的构建，Retrofit的优点再于通过接口的形式定义Http请求的信息，并且它支持所有的请求方式、和参数配置情况，能够很好的支持REST接口。Retrofit的设计是插件话的，它做的工作仅仅是通过动态代理和注解的方式来帮助生成实际对应某个接口Http请求调用，而至于具体使用的Http请求库如OkHttp,Json解析库，都是提供方法让用户自行选择配置的。

本文基于Retrofit+RxJava+OkHttp+Gson来搭建网络接口数据访问和解析层，Retrofit与OkHttp用于解析接口参数和发起网络请求，Gson用于将返回的Json数据直接转换成对应的JavaBean，RxJava用于将网络请求方法转换成Observabe，使得网络请求的调用也能用函数式编程的方式来使用。具体步骤如下:

1. build.gradle中配置这些支持库  
   compile 'com.squareup.retrofit:retrofit:2.0.0-beta2'  
   compile 'com.squareup.retrofit:converter-gson:2.0.0-beta2'  
   compile 'com.squareup.retrofit:adapter-rxjava:2.0.0-beta2'  
   compile 'com.squareup.okhttp:okhttp:2.7.0'  
   compile 'com.squareup.okhttp:okhttp-urlconnection:2.7.0'
2. 创建Retrofit.Builder，通过构建者模式配置这些参数。  
   Retrofit.Builder builder = new Retrofit.Builder();  
   builder.baseUrl("http://www.tngou.net/api/")  
    .client(okHttpClient)  
    .addCallAdapterFactory(RxJavaCallAdapterFactory.create())  
    .addConverterFactory(GsonConverterFactory.create())  
    .build();
3. 创建一个interface类，定义项目中用的所有请求Api，部分代码如下:  
   public interface TinaGouApi {  
    @GET("info/list")  
    Observable<HealthInforData> getHealthInforData(@Query("id")int classifyId, @Query("page")int page);  
   }  
   这里的@Get(“info/list”)表示用Get方式请求，info/list表示请求的接口是baseUrl+info/list，@Query(“id”)表示get方法请求的参数key为id，值为方法的参数。Retrofit还有很多注解的使用，可在官方文档查询。
4. 定义好请求方法接口后，使用之前创建的Retrofit.Builder对象调用build()方法创建Retrofit实例，之后调用Retrofit的create()方法，方法中传入之前创建的TianGouApi类，Retrofit会在运行时解析这个类的所有方法，动态代理调用方法。代码如下:  
   Retrofit ZousRetrofit = builder.build();  
   this.tinaGouApi = ZousRetrofit.create(TinaGouApi.class);

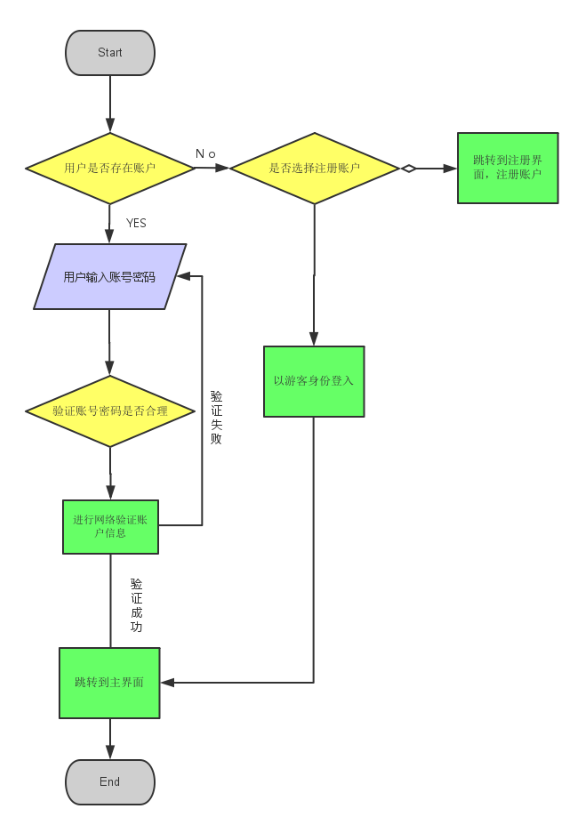
以上就完成了Http请求服务端的代码，在使用时，直接调用Retrofit.create()方法创建的TianGouApi实例即可。

# 系统功能模块的实现

## 登入注册模块的实现

### 登入注册模块的流程

由于程序评论的功能只有注册登入过的用户才能使用，没有注册的话需要进行账号注册，所以程序提供了登入界面来进行账户的登入。登入界面的基本流程如图



### 登入功能具体实现

登入功能的代码使用了MVP的架构模式，首先创建一个LoginContart接口类来将登入的视图和Presenter组织到一起，使用接口抽象定义登入模块视图和业务逻辑会拥有的行为,在登入界面视图层的逻辑包括forwardToMainView(跳转到主界面),forwardToRegister(跳转到注册界面),loginFailed(根据传入方法参数信息显示错误界面)，而Presenter层目前只有login（登入）方法，login方法根据传入的User信息访问服务端接口验证用户信息是否正确。

LoginContract抽象类

public interface LoginContract {  
 interface View extends BaseView<Presenter>{  
 void forwardToMainView();  
 void forwardToRegister();  
 void loginFailed(int meesageId,String message);  
 }  
 interface Presenter extends BasePresenter{  
 void login(User user, Context context);  
 }  
}

之后，创建具体的LoginContart.Presenter实现类LoginPresenter类。

LoginPresenter类

public class LoginPresenter implements LoginContract.Presenter {  
 LoginContract.View mLoginView;  
 public LoginPresenter(LoginContract.View view) {  
 mLoginView = view;  
 }  
  
 @Override  
 public void login(User user, Context context) {  
 user.login(context, new SaveListener() {  
 @Override  
 public void onSuccess() {  
 mLoginView.forwardToMainView();  
 }  
  
 @Override  
 public void onFailure(int i, String s) {  
 //todo 行为记录分析  
 mLoginView.showLoginFaield(i,s);  
 }  
 });  
 }  
  
 @Override  
 public void subscribe() {  
  
 }  
  
 @Override  
 public void unSubscribe() {  
  
 }  
}

　　定义好抽象的接口行为后，创建了LoginActivity作为具体视图载体，LoginAcitivty实现LoginContart的View接口，并在内部创建LoginContract.Presenter的具体实现类，只用调用setPresenter(Presenter p)方法将View也就是LoginActivity引用传给Presenter。此时，由于各个组件之间是通过抽象来绑定的，LoginPresenter只知道它绑定的视图时LoginView，而不关心LoginView的具体实现类是Acitivy还是Fragment，也不关心具体的先，这样做的好处是，当在后期的项目迭代过程中，如果Login视图类变化了不会影响到Login的业务逻辑部分。我们可以继续将视图界面的功能划分最小化，比如2个界面都拥有评论功能，可以抽象出一个评论相关的Presenter，而这个Presenter的实现类可以在不同的视图类中使用。

为了简化服务端代码的编写，服务端程序使用了Bmob云服务来搭建，用户对应的JavaBean包括以下这些字段

User类的部分字段的代码

private String username;// 用户登入名  
private String password;//密码  
private String email;//邮箱  
private Boolean emailVerified;//邮箱验证是否成功  
private String sessionToken;//session  
private String mobilePhoneNumber;//手机号  
private Boolean mobilePhoneNumberVerified;//手机号验

证是否成功

当用户输入完账号密码时点击登入按钮时调用atttemLogin()方法，attemLogin经过一些基本客户端验证，包括用户名密码是否过短，如果验证通过调用LoginPresenter的login方法进行Http请求网络验证。当验证通过时在回调方法中调用startActivity()方法跳转到MainActivity，如果网络验证失败则提示验证失败。部分代码如下

private void attemptLogin() {  
 …  
  
 // Store values at the time of the login attempt.  
 String email = mEmailView.getText().toString();  
 String password = mPasswordView.getText().toString();  
  
 boolean cancel = false;  
 View focusView = null;  
  
 // Check for a valid password, if the user entered one.  
 if (!TextUtils.isEmpty(password) && !isPasswordValid(password)) {  
 mPasswordView.setError(getString(R.string.error\_invalid\_password));  
 focusView = mPasswordView;  
 cancel = true;  
 }  
  
 // Check for a valid email address.  
 if (TextUtils.isEmpty(email)) {  
 mEmailView.setError(getString(R.string.error\_field\_required));  
 focusView = mEmailView;  
 cancel = true;  
 } else if (!isEmailValid(email)) {  
 mEmailView.setError(getString(R.string.error\_invalid\_email));  
 focusView = mEmailView;  
 cancel = true;  
 }  
  
 if (cancel) {  
 // There was an error; don't attempt login and focus the first  
 // form field with an error.  
 focusView.requestFocus();  
 } else {  
 // Show a progress spinner, and kick off a background task to  
 // perform the user login attempt.  
 showProgress(true);  
 User user = new User();  
 user.setUsername(mEmailView.getText().toString());  
 user.setPassword(mPasswordView.getText().toString());

mLoginPresenter.login(user, getBaseContext());  
 }  
}

当验证通过时跳转到主界面并将LoginActivity在Activity栈中移除，避免用户通过返回键时能跳转到登入界面造成困惑:

Intent intent = new Intent(this, MainActivity.class);  
startActivity(intent);  
finish();

验证失败时隐藏进度条通过Toast提示用户验证失败：

public void loginFailed(int meesageId, String message) {  
 showProgress(false);  
 Toast.makeText(this, "密码验证失败", Toast.LENGTH\_LONG).show();  
}

### 注册功能实现

　　注册功能的逻辑是用户输入需要注册的账号密码，访问后端接口注册。注册功能的界面和代码与登入界面基本相同，登入界面是验证后端数据库中该账号是否存在密码是否正确，注册界面是验证账号是否已被注册，其他功能和代码基本相同。

## 健康资讯模块的实现

健康资讯是用户进入到主界面时，默认显示的信息界面，将健康资讯分为4类数据来展示，分别是“生活贴士”、“药品新闻”、“食品新闻”、“疾病快讯”。使用TabLayout和ViewPager组合来展示内容，每一列表的都是由Fragment组成的，当点击列表中的一项时，跳转到健康资讯详情界面，健康资讯详情界面的UI实现控件也比较简单，都是最基本的Android原生控件搭建而成。

在健康资讯详情界面，除了基本的资讯讯息信息展示外，还有收藏该条资讯和对该资讯进行评论的功能。健康资讯模块的UI实现运行效果界面如图所示



资讯详情界面的帖子收藏功能基本逻辑是判断是否已经收藏，没有收藏则收藏，即将该条数据存入到收藏的数据库表中。 评论帖子的功能，是将该帖子的id和评论内容上传到服务器的评论表中。

收藏帖子的方法

public static boolean collectHelathInfor(HealthInfor healthInfor) {  
 CollectedHealthInfor collectedHealthInfor = new CollectedHealthInfor();  
 collectedHealthInfor.id = healthInfor.getId();  
 collectedHealthInfor.healthInfor = healthInfor;  
 long save = App.sDb.save(collectedHealthInfor);  
 return save > 0;  
}

评论帖子的方法代码

public void subMitRecommend() {  
 String recommendStr = recommendEtv.getText().toString();  
 HealthComment healthComment = new HealthComment();  
 healthComment.postId = inforId;  
 healthComment.comment = recommendStr;  
 BmobUser currentUser = User.getCurrentUser(this);  
 if (currentUser != null) {  
 healthComment.UserName = currentUser.getUsername();  
 } else {  
 healthComment.UserName = "匿名者";  
 }  
 healthComment.save(this, new SaveListener() {  
 @Override  
 public void onSuccess() {  
 Log.w("logger", "上传成功");  
 recommendEtv.setText("");  
 mCommentAdapter.add(healthComment);  
 mCommentRecyclerView.scrollToPosition(mCommentAdapter.getItemCount());  
 Toast.makeText(getBaseContext(), "评论成功", Toast.LENGTH\_SHORT).show();  
 }  
  
 @Override  
 public void onFailure(int i, String s) {  
 Log.w("logger", "上传失败");  
 }  
 });  
}

## 疾病库模块的实现

　　疾病库模块的设计是当用户在侧滑菜单点击“查一查”菜单时，进入疾病药品搜索系统，用户点击疾病或者药品图标时弹出一个Dialog，在Dialog的输入框中用户输入关键词进行查询。当查询得到结果时，如果是疾病跳转到DieseaseActivity界面显示该疾病的详细信息。如果是药品时，如果关键词信息较少结果通常是多个，使用一个LinearLayoutManager的RecyclerView来显示所有的药品，当用户点击某一项药品时，跳转到详情界面显示该药品的详细信息。

## 附近医院模块的实现

### 附近医院模块设计

附近医院模块的设计上当用户点击菜单并进入附近医院界面时，程序能够直接定位到用户当前所在的位置，用户通过菜单选择显示附近医院，能够显示出周边一定范围内的医院。周边医院的显示需要在地图上用指示物标注出来，当用户在地图点击时，显示出该医院的详细信息。

### 集成百度SDK

附近医院模块使用了百度地图SDK,百度地图的官网有Android Studio开发环境集成指南 <http://lbsyun.baidu.com/index.php?title=androidsdk/guide/buildproject>

### 使用方位传感器

用户在行走时，地图上应该用一个指示坐标显示出当前用户移动的方位，Android系统提供了方位传感器使得我们可以获得手机目前移动的方位。

方位传感器数据的使用方法是，创建一个Listener类实现SensorEventListener暴露的两个回调方法 onSensorChanged(SensorEvent event)和onAccuracyChanged(Sensor sensor, int accuracy)，由于我们只需要Sensor方位的信息，所以只需要实现在onSensorChanged，通过判断SensorManager.DATA\_X的差值，然后将差值回调给自己实现的回调接口即可。

回调方法onSensorChanged的具体实现

public void onSensorChanged(SensorEvent event) {  
 if ( event.sensor.getType() == Sensor.TYPE\_ORIENTATION){//如果是方向传感器的话  
 float x = event.values[SensorManager.DATA\_X];  
  
 if (Math.abs(x - lastX) >1.0){  
 //数值回调给监听接口  
 if (mOnOrientationListener != null)  
 {  
 mOnOrientationListener.onOrientationChanged(x);  
 }  
 }  
 lastX = x;  
 }  
}

### 附近医院功能的具体实现

有关地图添加覆盖物、罗盘模式、定位方位等功能的实现都在BaiduMap的文档中有详细的例子。当通过网络请求获得用户附近的医院信息后，通过BaiduMap Api根据具体的方位信息，在地图上添加标注物，此时当用户点击标注物后，调用searchPoiDetai()获取点击位置对应的医院信息，根据获得的医院详情，修改并显示原本隐藏的用于显示医院详情的布局。代码如下

mBaiduMap.setOnMarkerClickListener(new BaiduMap.OnMarkerClickListener() {  
 @Override  
 public boolean onMarkerClick(Marker marker) {  
 Bundle extraInfo = marker.getExtraInfo();  
 //搜索Marker的详细信息  
 mPoiSearch.searchPoiDetail(new PoiDetailSearchOption().poiUid(marker.getExtraInfo().getString("uid")));  
 mMarkerLayout.setVisibility(View.VISIBLE);  
 //为点击Marker 的位置处显示inforWindow  
 return true;  
 }  
});

该模块功能的运行界面效果如下图所示：





## 健康问答模块的实现

健康问答的类别较多，当用户在侧滑菜单点击该功能时，首先通过GridLayout展示所有的健康问答类别，当用户点击具体的类别时，显示出该类别下的所有问答信息。健康问答模块的设计基本与健康资讯相同，区别在于健康资讯类别数量较少，类别的分类直接在Acitivity中使用TabLayout来展示，健康问答由于类别较多，使用了单独的Fragment包含一个RecyclerView来展示。以下是健康问答模块的分类界面的UI视图。



## 其他模块

除了以上的主要模块，程序中在菜单界面还添加了一些常见应用的基本功能，比如包含清除缓存、定时提醒等功能的设置界面，还有分享应用和有关该程序接好的关于界面。

# 系统关键问题及解决

## 多级缓存的实现

为了提高App的体验，视图层在获取数据的基本流程如下

网络请求使用了OkHttpp Client来发起，对于网络请求的结果进行本地缓存，

### 开启相应数据缓存到文件系统功能

系统使用OkHttp 来进行网络请求，默认情况下OkHttp不会缓存响应数据。这会导致客户端浪费时间和流量去多次请求通用的数据。为了开启缓存响应数据功能，需要先创建一个 com.squareup.okhttp.Cache 对象并将其作为参数传给OKHttpClient的setCache方法。代码如下：

okHttpClient.setCache(new Cache(cacheFile, 1024 \* 1024 \* 10));  
　　Cache的构造函数的2个参数分别表示，数据存储的本地路径和，数据存储的大小。

### Fresco配置图片缓存

Fresco的缓存一共分为3级，分别为Bitmap缓存、未解码图片的内存缓存、文件缓存。Fresco的缓存是由ImagePipeline类来控制的，在配置文件缓冲时可以控制是使用一个文件缓存或者两个文件缓存，调用setMainDiskCacheConfig和setSmallImageDiskCacheConfig即可。ImagePieline还有非常多的可配置的行为，通过ImagePipelineConfig来定义，下面是ImagePieline的所有可配置选项。

ImagePipelineConfig config = ImagePipelineConfig.newBuilder()

.setBitmapMemoryCacheParamsSupplier(bitmapCacheParamsSupplier)

.setCacheKeyFactory(cacheKeyFactory)

.setEncodedMemoryCacheParamsSupplier(encodedCacheParamsSupplier)

.setExecutorSupplier(executorSupplier)

.setImageCacheStatsTracker(imageCacheStatsTracker)

.setMainDiskCacheConfig(mainDiskCacheConfig)

.setMemoryTrimmableRegistry(memoryTrimmableRegistry)

.setNetworkFetchProducer(networkFetchProducer)

.setPoolFactory(poolFactory)

.setProgressiveJpegConfig(progressiveJpegConfig)

.setRequestListeners(requestListeners)

.setSmallImageDiskCacheConfig(smallImageDiskCacheConfig)

.build();

Fresco.initialize(context, config);

配置好的ImagePielineConfig需要传递给Fresco.initialize用于初始化Fresco。

## 使用RecyclerView替换ListView

RecyclerView是Android Support v7包提供的用于替换ListView和GridView的控件，ListView和GridView的设计基本相同，区别仅仅在于视图的排列方式不同，于是RecyclerView将对itemView的排列方式单独抽取出，抽象出LayoutManager用于指定itemView的排列方式。RecyclerView只负责itemView的回收与服用，其他的自己来实现，实现了高度的解耦。

在本文项目中，大量使用了RecyclerView用于展示列表视图，下面介绍RecyclerView的使用方式

1. 在对应的布局文件中加入RecyclerView控件
2. 在代码中通过findViewById(()得到控件后，调用setLayoutManager()方法指定视图的布局方式
3. 创建需要展示的数据的对应的RecyclerView.Adapter子类,并实例化。调用RecyclerView.setAdapter(Adapter adapter)将实例化的对象入参。

### RecyclerView的优点及使用

### 简单封装通用的Adapter

以往在使用ListView时，为了减少ItemView的findViewById()方法的调用，会在Adapter内部创建一个ViewHolder用于存储View的引用以减少方法的调用，提高效率。RecyclerView.Adapter默认并强制用户使用ViewHolder的模式来提高效率，在本项目中，使用到RecyclerView.Adapter的地方较多，而且逻辑基本相同，区别仅仅在于布局文件和视图数据的不同，为了简化代码的编写，在RecyclerView.Adapter的基础上封装一个通用的BaseAdatper是很有用处的。基本的思想步骤如下.。

1. 创建一个SuperViewHolder类继承RecyclerView.ViewHolder，在内部使用SpareArray创建一个集合，SpareseArray是android.util包下一个优化的以int为键的Map。使用SparseArray以View的id为键，View对象为值，存储视图对象。类内部创建一个的getView(int id)方法，方法的参数id，表示View的id，当判断该id在spareseArray中不存在该键值时，调用findViewById()方法查找该View，如果找到该View，则将该View存入SparseArray，并将它最为方法返回值return。
2. 创建BaseSuperAdapter抽象类继承RecyclerView.Adapter<SuperAdapter>，此时需要实现getItemCount()、onBindViewHolder()、onCreateViewHolder()三个抽象方法。BaseSuperAdapter的构造函数需要让用户传入一个ListView集合，作为Adapter的View对应的ViewData，getItemCount()方法返回集合的长度，onCreateViewHolder()直接创建SuperViewHolder。onBindViewHolder ()方法是系统第一次创建ViewHolder或者复用ViewHolder时回调的，在方法内部，我们通过方法的position参数获得集合该位置对应的数据，并创建一个抽象方法onBind(ViewHolder holder,T item),将holder和对应的数据传入。

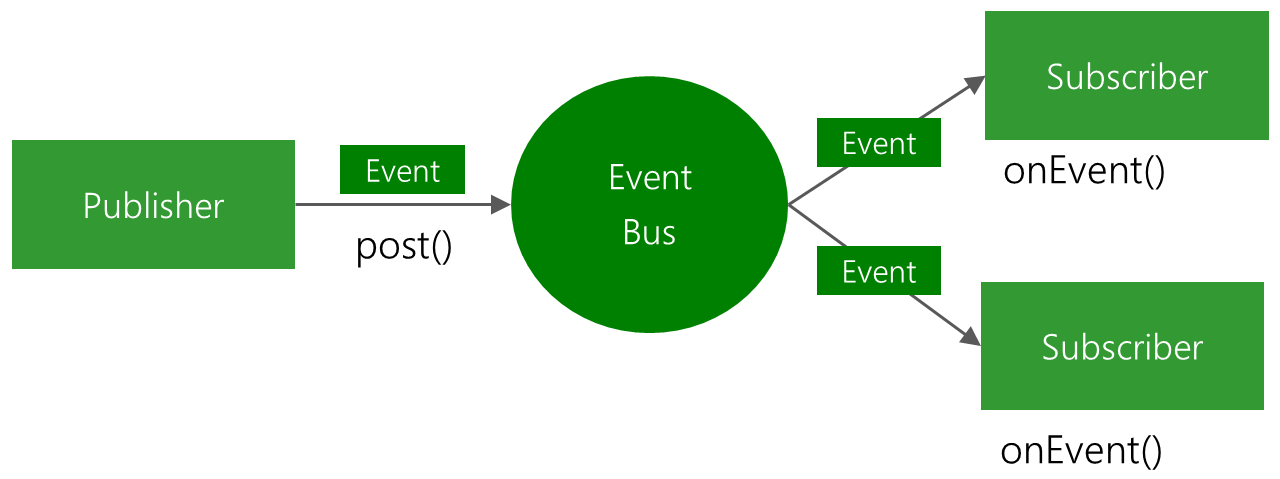
在使用时，我们直接继承SuperAdapter，此时只需要实现构造函数和onBind()抽象方法，大大简化了Adapter的编写。比如本项目中评论列表使用的Adapter代码如下。

public class CommentAdapter extends SuperAdapter<HealthComment>{  
 public CommentAdapter(Context context, List<HealthComment> items, int layoutResId) {  
 super(context, items, layoutResId);  
 }  
  
 @Override  
 public void onBind(SuperViewHolder holder, int viewType, int layoutPosition, HealthComment item) {  
 holder.setText(R.id.id\_recommend\_tv,item.comment);  
 }  
}

## 基于RxJava实现事件通信功能

在Android项目的开发过程中，在不同界面的事件触发的整个流程偶尔会贯穿几个界面，在不同的Android组件中进行通知事件的发生，一般会采用2种方式，一是采用接口回调的形式，比如最常见的OnClickListener，这种方式的缺点在于在复杂场景中回调过多时容易陷入“回调地狱”，代码较多耦合；另一种方式是实现一个事件通信总线，一个基本的事件通信框架的运行流程如图。

RxJava是基于观察者模式设计的编程框架，基于RxJava能够很简单的实现一个事件通信总线。



一个事件通信框架基本包含以下几个角色：Event、Subscriber、Publisher，Event的角色可以由RxJava中的Subject来充当，Subject默认是非线程安全的，Subject的子类SerializedSubject 是框架中提供的线程安全的Subject。另外，考虑到我们实际使用时，对于Subject已经发射的事件，Observer不再关心，所以需要在SerializedSubject的构造函数中传入PublishSubject对象，PublishSubject不会再内部存储已经发射过的事件，当观察者订阅时，他只发射在订阅发生之后的事件。

Event的角色就是不同的事件，Event根据我们的项目事件情况，需要我们自己创建。Subscriber的角色创建比较灵活，RxJava提供了多种创建Subscriber的方式。

发射事件的方式也很简单，subject的onNext(Object)方法，会将参数中的对象传递给观察者。订阅事件的方式通过将内部的Subject对象转换成Observable,Observable有subscribe()方法即可实现订阅。

本文实现的事件通信类如下所示:  
public class RxBus {

//考虑到多线程调用情况，建议使用serializedSubject包装原有的Subject对象,它使用了AtomicReference保证了多线程操作的原子性...

//使用PublishSubject.create()创建 PublishSubject，

// 当Observer订阅Subject时，它只会释放订阅之后来自原始Observable发射的数据

private final Subject<Object, Object> bus = new SerializedSubject<>(PublishSubject.create());

private RxBus() {

}

public static RxBus getDefault() {

return Singleton.rxBus;

}

public void send(Object event) {

bus.onNext(event);

}

public Observable<Object> toObservable() {

return bus;

}

/\*\*

\* 泛型 返回指定类型的Observable,内部通过判断 instanceof 来进行过滤

\*/

public <T> Observable<T> toObservable(final Class<T> type) {

return bus.ofType(type);

}

public <T> Subscription register(final Class<T> type, Observer<T> observer) {

return toObservable(type).subscribe(observer);

}

public <T> Subscription register(final Class<T> type, Action1<T> action1) {

return toObservable(type).subscribe(action1);

}

public boolean hasObservers() {

return bus.hasObservers();

}

static class Singleton {

public static RxBus rxBus = new RxBus();

}

}

# 总结与展望

## 论文总结

## 展望

# 参考文献

1. 何华刚, 裴先明．化工安全评价探讨[J]．安全与环境工程, 2003, 10(1): 56-59．
2. 马海锐．化工安全评价[J]．科技资讯, 2011, (24): 174-175．
3. 蔡琢．化工园区区域风险综合评价技术研究[D]．南京：南京工业大学, 2008．
4. 李杰．炼油石化企业安全风险评价研究[D]．天津：天津大学, 2010．
5. 蔡文, 杨春燕, 王光华．一门新的交叉学科——可拓学[J]．中国科学基金, 2004(5): 268-272．
6. 孙佰清, 刑爱国, 张积宾, 潘启树．可拓神经网络模型的设计与实现[J]．哈尔滨工业大学学报, 2006, 38(7): 1156-1159, 1191．
7. 张玉斌, 张云辉, 李磊, 滕春贤．基于可拓方法的上市公司综合评价模型[J]．哈尔滨理工大学学报, 2003, 8(4): 135-138．
8. 周开君．基于可拓学的航道水域通航环境安全评价研究[D]．大连：大连海事大学, 2011．
9. 蔡文, 石勇．可拓学的科学意义与未来发展[J]．哈尔滨工业大学学报, 2006, 38(7): 1079-1086．
10. 王洪伟．可拓学的进展及应用研究[J]．广东工业大学学报, 1998, 15:111-116．
11. 徐园．可拓理论在过程工业中的应用研究[D]．北京: 北京化工大学, 2010．
12. 匡乐红, 徐林荣, 刘宝琛．基于可拓方法的泥石流危险性评价[J]．中国铁道科学, 2006, 27(5): 1-5．
13. Wen Cai．Extension theory and its application[J]．Chinese Science Bulletin, 1999, 44(17): 1538-1548．
14. CAI Wen．Extension management engineering and aplications[J]．International Journal of Operations and Quantitative Management, 1999(1)：59-72．
15. Meng-Hui Wang, Yu-Kuo Chung, Wen-Tsai Sung．The Fault Diagnosis of Analog Circuits Based on[J]．Department of Electrical Engineering National Chin-Yi University of Technology, 2009, 735-744．

# 致 谢

本论文在刘启玉老师的悉心指导下得以顺利完成。从论文的选题、评价方法和主要内容的研究直至论文的最终完成，都受到了刘老师耐心细致的指导，他总是细心地帮我检查论文，指导我修改论文，使我从中受益匪浅。历时四载，刘老师一丝不苟的作风，严谨求实、认真负责的态度，不仅授我以文，那种敬业精神更是令我敬佩。

在此，我真诚感谢诸位老师四年来的教导与帮助，你们的谆谆教诲指引着我走好每一步；感谢和我朝夕相伴的同学，是你们陪伴我度过了轻松快乐且令人终生难忘的大学四年时光；感谢我的寝室室友，谢谢你们四年来带给我的帮助和欢笑、支持与鼓励。