

Android应用开发综合实验报告

课题名称： 基于Android的天气预报APP

学 院： 计算机与信息安全学院

成 员： 1900301113龚俊源

1900300305梁嘉桐

1900301027刘 航

报告日期： 2022年 6月 5日

表1 课程设计团队任务分工表

|  |  |  |  |
| --- | --- | --- | --- |
| 题目 | 基于Android的天气预报APP | | |
| 序号 | 学号 | 姓名 | 个人任务描述 |
| 1 | 1900301113 | 龚俊源 | Android端开发 |
| 2 | 1900300305 | 梁嘉桐 | 需求及产品设计，UI及UX交互设计 |
| 3 | 1900301027 | 刘航 | 后台接口开发 |

**前 言**

随着科学技术的发展，互联网几乎存在于我们每个人的生活中，而智能手机变成了我们工作、生活和学习中必不可少的设备。现在慢慢进入5G时代，人们追求的速度也在不断得到了满足，而使用的app也要不断优化改进从而满足人们的各种需要。根据目前最新的数据统计，Android用户的数量全球范围内来说占比是最高的，同样该系统也是一个真正意义上的开发性综合平台，能够允许所有的机构或者个人进行开发，是一个能够满足学生对于知识进行实践的操作系统。作为新时代的青年人，不能仅仅关注于自身的学习还有生活，还要关心社会和国家的动态。在互联网迅猛发展的今天，天气预报app几乎已经成为了每个人手机当中不可或缺的工具，通过这款app去了解各个时段天气的变化情况，并根据天气情况进行增衣减衣。我们开发天气预报app的目的是能够及时查询各个城市的天气情况，通过一个简易的视图结构，将温度、天气情况、湿度等等展示出来，方便每个使用该款天气预报app的用户。

**关键词：** Android开发、云端数据库应用、JAVA

目 录

[一、概要设计 1](#_Toc105571465)

[1.1需求描述 1](#_Toc105571466)

[1.2整体开发设计架构 1](#_Toc105571467)

[二、详细设计 2](#_Toc105571468)

[2.1页面功能 2](#_Toc105571469)

[2.2功能实现 3](#_Toc105571470)

[2.3逻辑实现 3](#_Toc105571471)

[三、软件测试 5](#_Toc105571472)

[3.1用户注册测试 5](#_Toc105571473)

[3.2用户登录测试 8](#_Toc105571474)

[3.3 App主页面新闻内容显示测试 8](#_Toc105571475)

[3.4添加城市信息搜索测试 10](#_Toc105571476)

[3.5删除添加城市信息测试 16](#_Toc105571477)

[3.6清除app所有缓存测试 18](#_Toc105571478)

[3.7退出账户登录的测试 19](#_Toc105571479)

[3.8 App分享的测试 22](#_Toc105571480)

[四、参考文献 24](#_Toc105571481)

[五、心得体会 25](#_Toc105571482)

[六、附录 27](#_Toc105571483)

[6.1用户登录 27](#_Toc105571484)

[6.2天气信息的加载 28](#_Toc105571485)

[6.3用户找回/修改密码 29](#_Toc105571486)

[6.4发送验证码 31](#_Toc105571487)

[6.5主页天气Adapter 33](#_Toc105571488)

[6.6管理城市界面Adapter 35](#_Toc105571489)

[6.7删除城市Adapter 36](#_Toc105571490)

一、概要设计

1.1需求描述

在过去人们只能通过报纸、新闻广播、观看电视的方式来获取天气情况，并且不能获得实时的天气情况，往往这样的方式是具有一定的时延性，人们不能及时获取消息，突如其来的雨雪天气使人们的出行不便。目前我们正处在一个5G即将普及的信息时代，对于信息获取的时效性有了更高的要求，用户往往希望能够更快速获取实时的信息。在互联网盛行的时代，天气预报的实时性是必要的，如果像以前一样的方式想要获取实时天气情况，是不可行的，天气预报的准确性也缺少了某时刻的价值，用户缺少了自主查询权，如遇糟糕天气，便会影响一定的出行。一款可以直接在手机上查询实时天气预报的app就能够满足用户当前对于时效性、自主选择性的需求。

1.2整体开发设计架构

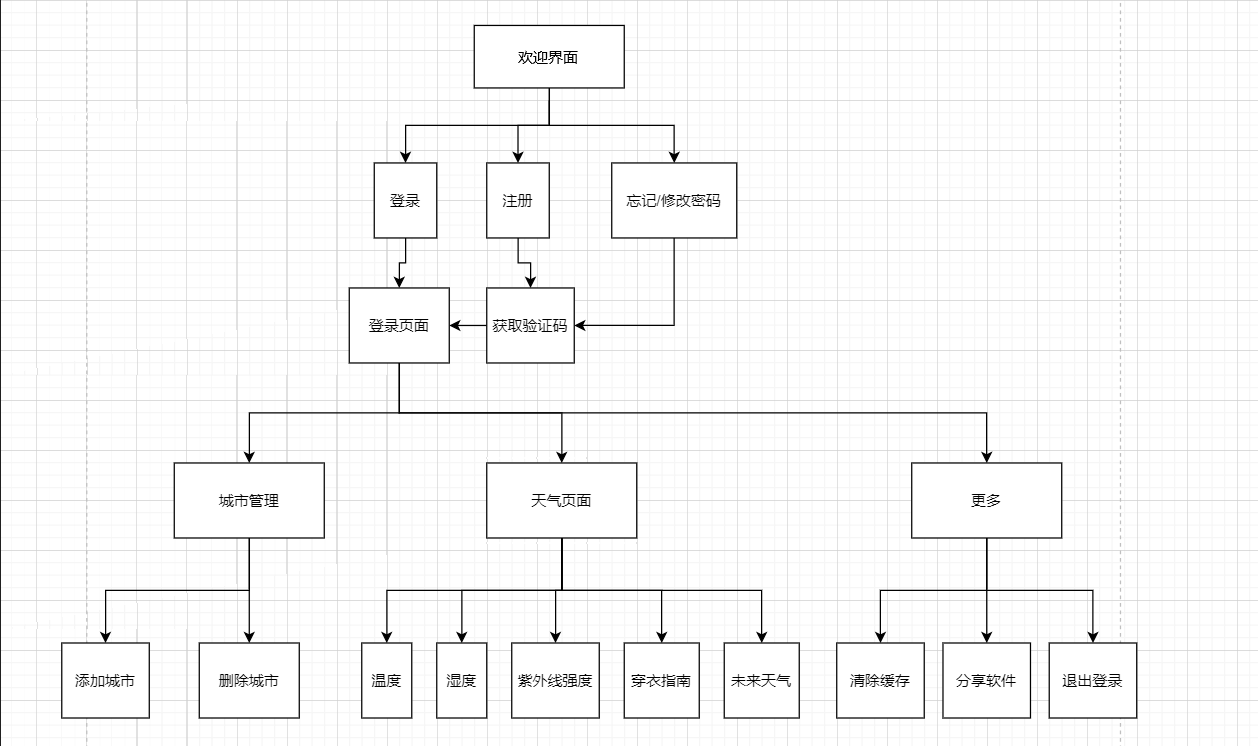
****

图1.app整体框架

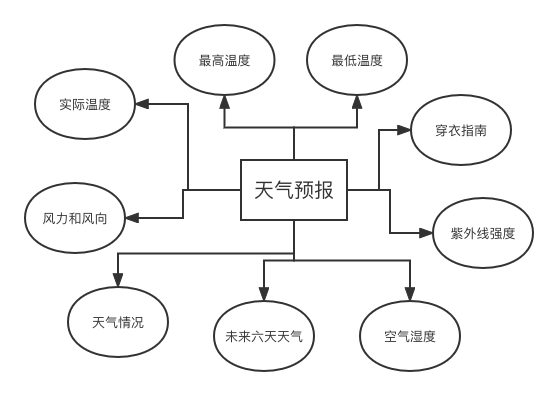
****

图2.天气预报E-R图

二、详细设计

2.1页面功能

本应用包含登陆页面、主页面、城市管理页面、重置管理页面、更多设置页面、搜索引擎页面。打开虚拟机时，会看到一个“guet”图标，点击图标可以进入该app的登录界面，在登录界面可以通过手机验证码的方式去进行账号注册和账号密码的修改，登录之后可以进入app的主页面。在应用开启时连网便会在主页面显示此时此刻的天气情况，如果没有添加城市，便会出现默认南宁城市天气情况，包含最高最低温度、湿度、天气情况、风度、日期、穿衣指数、紫外线强度。主页面第一行显示当前温度，下方显示未来6天的天气信息，左下角为重置管理页面，可以实现自行添加城市，如需添加跳转城市管理页面，搜索界面下方包含几大城市，如需搜索别的城市也可自行输入搜索，搜索引擎在输入不存在的城市时，系统跳出提醒“没有收录该城市信息”，输入为空时会跳出提醒“输入不能为空”，当输入国外城市时，会跳出提醒“没有城市信息”，因为该系统只支持国内查询天气情况。重置管理页面内包含删除城市的按键，可进行增删操作。当本页面删减空时，按确定键后系统自动添加南宁城市的天气情况，并在主页面显示。更多设置页面显示当前版本、清除缓存、分享软件、退出登陆功能。清除缓存功能会删除所有信息，包含城市管理页面内的信息，当城市管理页面内的城市信息全部删除后，会自动添加南宁城市的天气信息。

2.2功能实现

登录页面实现：包含账号/密码，注册和修改密码界面。登录界面由Check、FindPassword、Login、Register、User\_Table几个类组成，分别实现账号密码的检测、账号密码的输入状态正确与否、登录、注册功能实现和返回用户账号密码的功能。

主页面实现：包含城市管理按钮和更多按钮。页面用viewpager实现，activity里套用fragment，用fragment实现主页面程序切换功能，未来六天天气情况在item- fragment- center内插入，其中包含天气、温度等信息。

城市管理页面实现：程序管理里的城市用listview和cardview实现，listvie列表内的每一个项用在city-manager内实现编辑，city-manager内编辑的是每一个城市管理的样貌。

搜索引擎界面：用网格视图实现，搜索的东西只是个名称所以这里没有用listview实现，只需用数组的适配器适配就可得到结果，

重置管理页面实现：用listview和cardview实现，每个item用cardview实现，热门城市添加在网格视图内，里面可编辑底色等信息。

2.3逻辑实现

主页面：最开始建立基类mainactivity，用xutils加载网络数据，声明整体模块，执行网络请求，请求得到天气预报数据的api，cityweatherfragment请求数据用basefragment继承，basefragment请求数据继承xutils。在basefragment内构建loaddata，loaddata为请求数据的方法，通过url的地址来进行get的数据，网络请求参数实体requestparams，其中url不可为空，builder用于自定义参数构建过程，为空时使用@link defoultparamsbuilder，signs自定义需要签名的字段，传入paramsbuilder，cachekeys自定义缓存key信息，传入paramsbuilder。 使用xutils的第一步要先创建自己的application类，所有的任务控制中心，http，image，数据库，view注入等接口的入口全在其中，在oncreate内进行初始化以及生成database全局对象数据库；wetherbean类用于储存网络传输请求回来的数据，类似信息中转站。

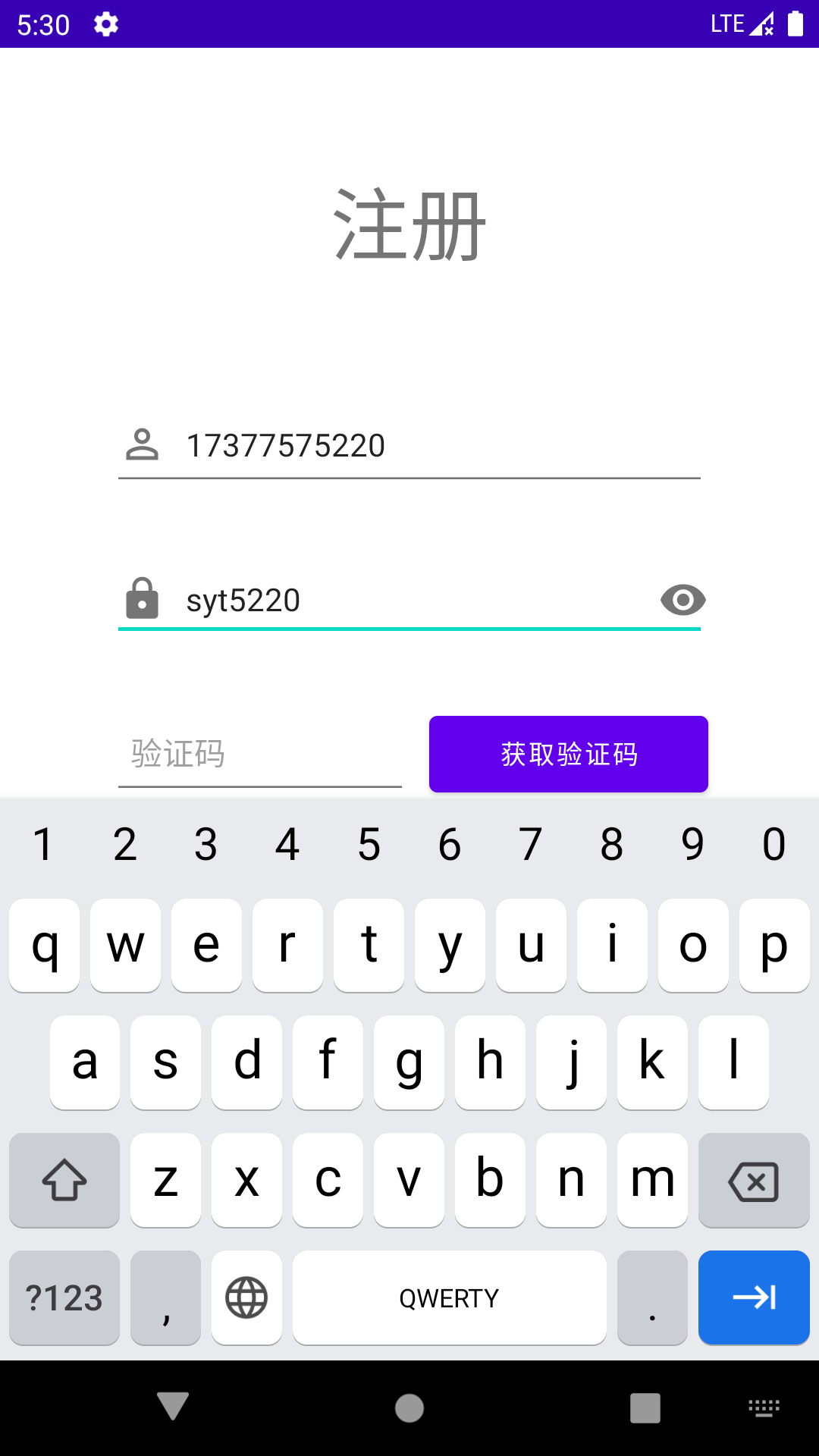
组键层：添加城市，定义内容。用addcityiv形成主页面左下脚的设置按钮，用moreiv实现更多按钮，用pointlavout实现页面下方的小点，用vierpager mainvp实现主页面的主内容，vainpager的数据源为fragmentlist，显示城市的集合用citylist实现，viewpager的页面指数器用imglist实现，用adapter定义fragment的适配器，创建内容，添加点击事件，用fragmentlist=new arraylist定义一个列表，citylist从数据库里获取，为数据库内包含的城市，当citylistsize为0即没有数据就增加南宁一个城市。初始化viewpageer，通过创建fragment对象，添加到view'pager数据源中，通过bundle显示的城市列表传入bundle，通过bundle设置fragment的参数，将activity的内部值传给对应的fragment，在对应的fragment内接受参数，添加数据并通过fragmentlist构建一个适配器，用自动生成的getsupportfragmentmanager方法设置在适配器的内容。页面通过fragmentmanager自动生成getitem。动态创建小圆点，有多少个页面（城市信息）就创建多少个小圆点，将内容加入pointlayout内，pointlayout内包含对小点点的定义。实现小点点自动切换功能需要在页面内添加一个监听器，当页面转换时，小点点变绿，而不是灰色的。viewpager页面监听函数，在onpageselecte被选择的页面所在位置，先全部设为灰色，再把当前位置设置成绿色。用mainvp setcurentitem显示最后（最新）一个城市表示的信息。

城市管理页面：有三个适配器，search内包含热门城市列表，此处时自定义，其中包含请求的地址api，通过传入hotcitys和arrayadapter适配器显示出来数据。当点击每个热门城市时都会自己跳转该城市的天气状况。用textutils isempty（city）判断用户是否在搜索时输入空字符以及能否找到该城市并进行相对应的跳转，传回去的数据会重写baseactivity内的内容，当调用成功后的数据会暂存成weatherbean类并存在weather类内，通过数据返回是否为200来判断是否成功，成功显示，否则显示暂未收录，城市存在，跳转回首页面，开一个新的栈，不考虑后退，后退则回到主页。把原来的内容全部清空，直接进入新栈跳转到mainactivity。设置特殊的flags来控制intent的处理事件，给intent附加一个标志，flage-activity-new-task清空栈内所有的activity，该标志必须和flag-activity-new-task一起使用。todo来传参数startactivity过来定义一个intent，把数据接收，并判断，如果不在城市列表中，就加到城市列表中，textutils isempty用于判断是否为null，citylist contains（city）用于判断city是否在城市列表。页面重新加载时调用，并更新viewpager，重复操作，创造新的页面，用notifydatasetchanged更新适配器。通过bean类定义的方法，以及setext获取天气数据，包含地址，风力，天气，温度，城市等，相当于构建getwin，再将其适配回去给adapter显示构造出来的适配器视图的内容，当启动activity时就会获取数据库中的真实数据源，并添加到现有数据源中，提示适配器更新。让listview显示各个item。本系统限制添加五个城市，按添加按钮添加后会自动跳转搜索页面，当城市超过五个后，会提示“存储上限”信息。删除界面的适配器通过view构造信息，这里不继承baseactivity，当删除后会显示适配器更新。设置监听事件erroriv，点击事件并在适配器内显示。delete出提醒信息，删除成功就会返回上一级页面。其中删除的信息是删除数据库中的内容。

三、软件测试

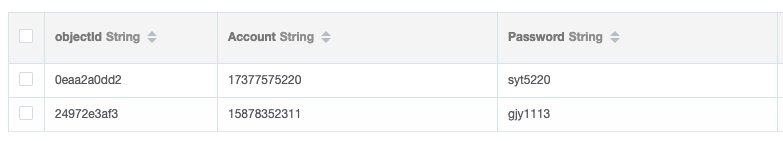
3.1用户注册测试

用户的账号为手机号，通过手机接收验证码来进行注册和修改密码功能的实现。





注册的用户信息可以通过bmob保存至云端，能够在云端查看到。



3.2用户登录测试

用户登录界面如图所示：



3.3 App主页面新闻内容显示测试

在app的主页面可以看到城市、天气、紫外线强度、湿度、温度、未来天气等信息。



默认初始城市设置为南宁，所以打开app城市管理界面可以看到只有南宁市一个城市信息。



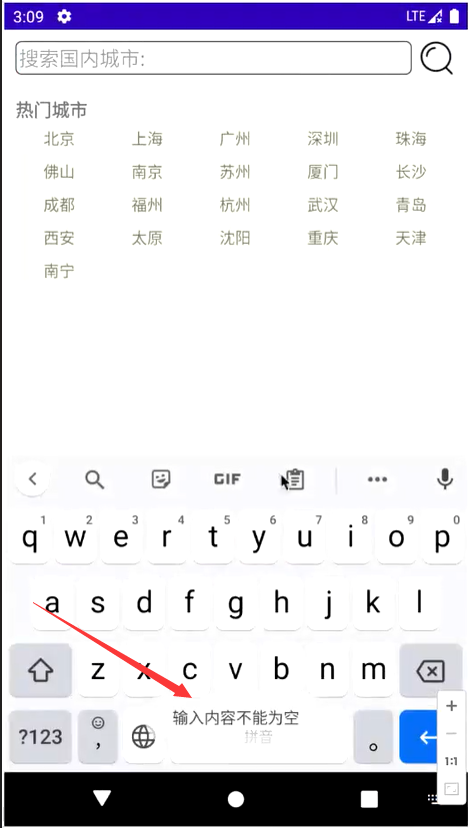
3.4添加城市信息搜索测试

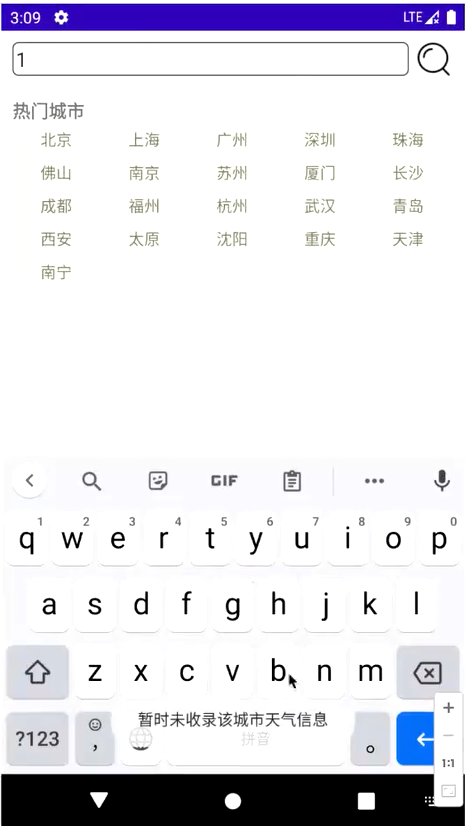




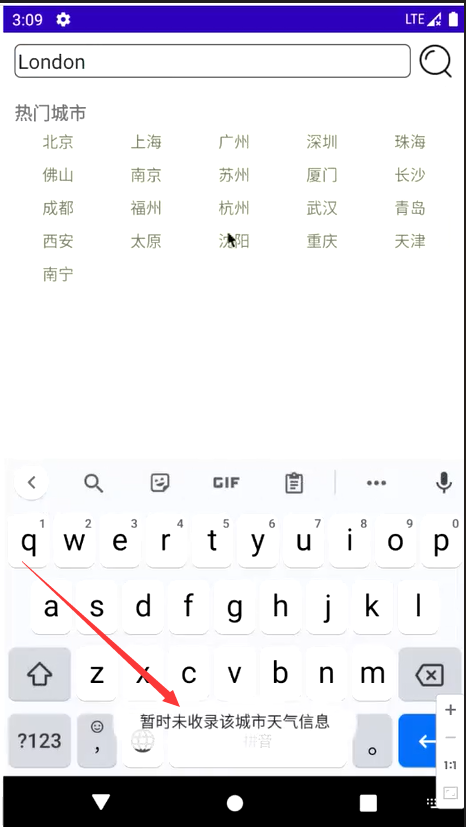


当输入空或者不存在的城市，会弹出出入内容不能为空的提示。



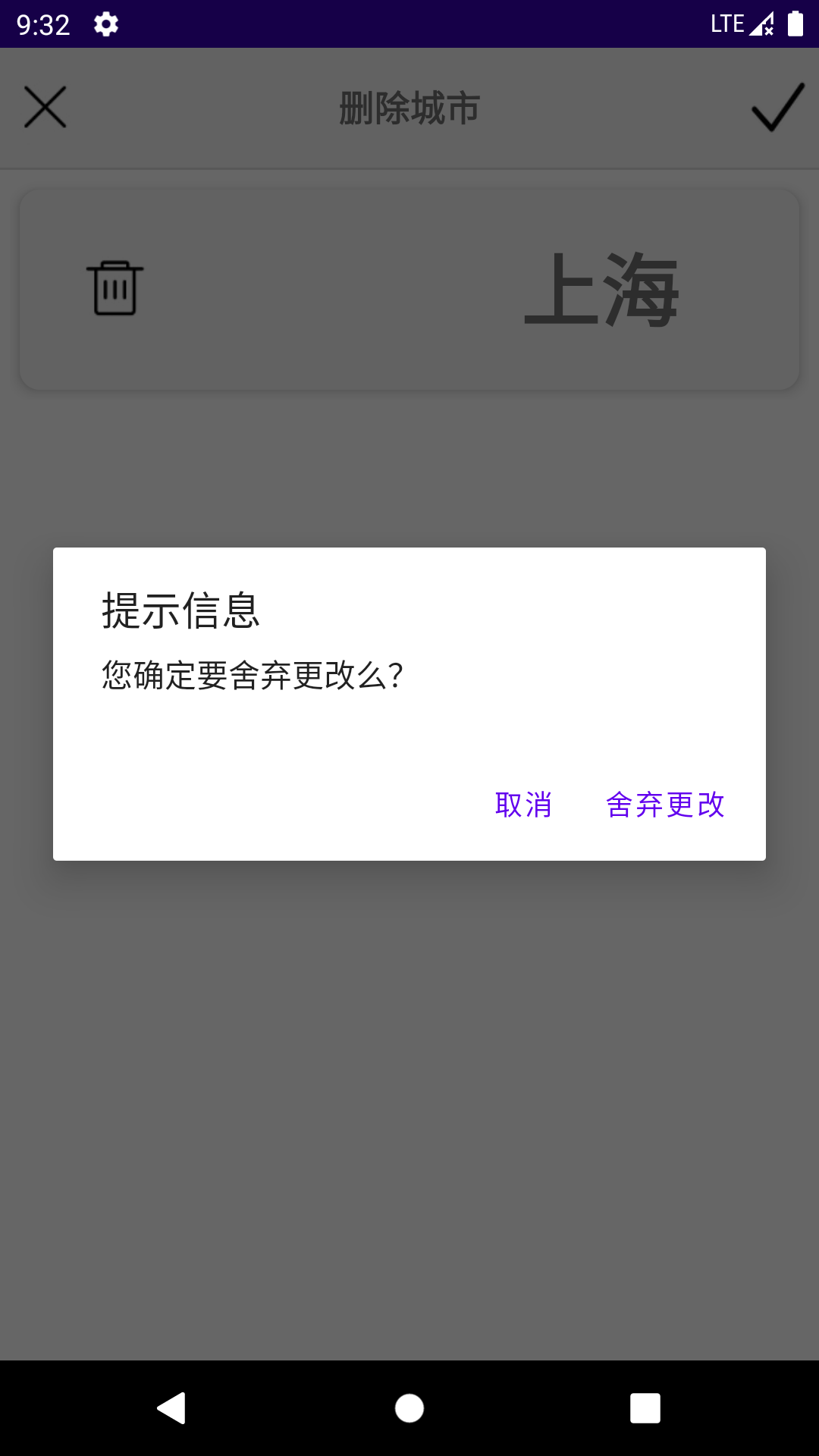


输入国外城市时，会显示未收录城市信息，因为只添加了国内的城市信息。



3.5删除添加城市信息测试





3.6清除app所有缓存测试



3.7退出账户登录的测试

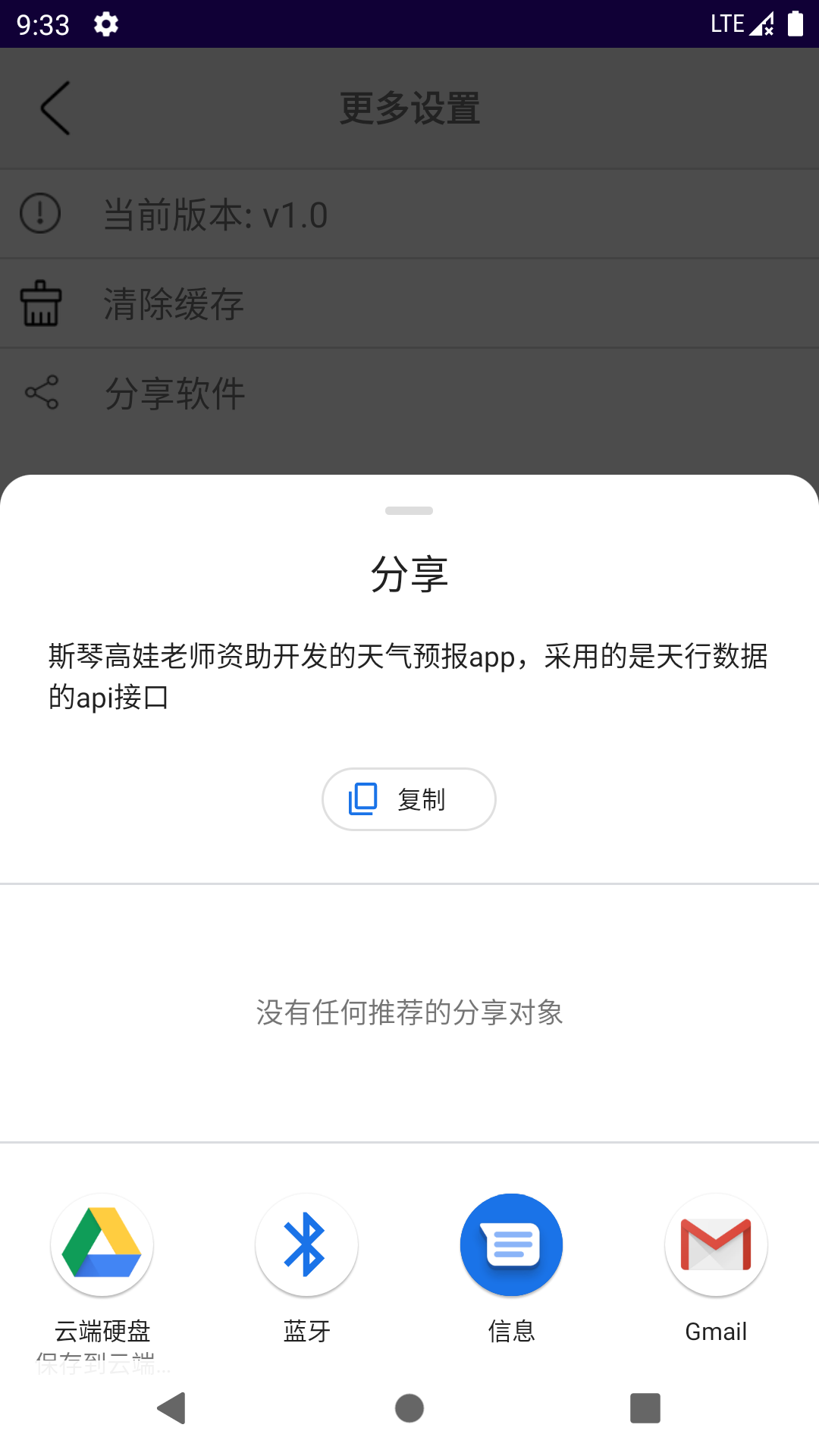
****



点击退出之后，重新回到登录界面。

****

3.8 App分享的测试

****

四、参考文献

[1] 郭霖.第一行代码——Android [M].北京：人民邮电出版社，2016：77~104，294~302.

[2] Muschko.B著;李建，杨柳，朱本威译.实战Gradle [M].北京：电子工业出版社，2015：1~20.

[3] 李刚.疯狂Android讲义（第三版） [M].北京：电子工业出版社，2015：35~247.

[4] 杨丰盛.Android应用开发揭秘 [M].北京：机械工业出版社，2010：188~208.

[5] Bmob官方 Bmob Android接口文档中心[EB/OL]. http://doc.bmob.cn/data/ android/index.html，2017.

[6] 施威铭. Android APP开发入门[M].北京:机械工业出版社,2016.

五、心得体会

在这次Android应用开发设计中，通过对基于Android的天气预报功能的实现，使我们对Android天气预报系统的整体设计有了更加深入的理解和认识，对整个开发流程也有了更清晰的认识。在开发过程中，不能急于求成。首先要对系统进行需求分析，明确系统要实现的功能之后再着手进行开发，这样在后续的编程过程才能更加顺利地进行下去。在进行具体的开发编码实现过程中，我们可以先进行原型设计，然后在这基础上再进行进一步的修改完善功能，或者是在开发之初，将开发过程分成几大主要部分，先抓住最核心的部分进行开发，然后围绕着这几大部分不断进行补充添加功能、完善设计。

在开发过程中我们学到了一些经验：系统分析的好坏将决定着的系统开发成功与否，一份好的分析设计将是成功开发的主要因素。我们在着手开发之前不要急于编程，应先有较长的时间去把分析做好，写出相关的开发文档等。然后再开始编写程序代码，这样做到每写一步代码心底有数,有条不絮。

本次设计开发是通过小组的形式进行的，我们小组的成员一开始都是小白，对于Android开发的过程以及一些技术都不是很熟悉，通过查阅相关资料和观看相关视频，最后了解了Android开发的流程，将新学的知识与之前已经学习过的知识进行整合，让我对软件开发的生命周期有了更加深刻的了解。通过小组成员的共同努力下，最后我们实现出了一个可以看到天气和未来几天天气的天气预报系统，不过最终实现出来的系统较为简单，系统功能不完善，后期仍然需要不断学习提高自己的专业水平，不断完善系统功能，使应用更加完美，做出一个更好更具有实用性的系统。

六、附录

6.1用户登录

// 登陆监听  
LoginButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 // 账号(Account)、密码(Password)  
 final String Account = AccountText.getText().toString().trim();  
 final String Password = PasswordText.getText().toString().trim();  
  
 if (TextUtils.isEmpty(Account)) {  
 Toast.makeText(Login.this, "请填写手机号码", Toast.LENGTH\_SHORT).show();  
 } else if (TextUtils.isEmpty(Password)) {  
 Toast.makeText(Login.this, "请填写密码", Toast.LENGTH\_SHORT).show();  
 } else {  
 BmobQuery<User\_Table> bmobQuery = new BmobQuery<>();  
 bmobQuery.findObjects(new FindListener<User\_Table>() {  
 @Override  
 public void done(List<User\_Table> object, BmobException e) {  
 if (e == null) {  
 //判断信号量，若查找结束count和object长度相等，则没有查找到该账号  
 int count=0;  
 for (User\_Table user\_table : object) {  
 if (user\_table.getAccount().equals(Account)) {  
 //已查找到该账号，检测密码是否正确  
 if (user\_table.getPassword().equals(Password)) {  
 //密码正确，跳转（Home是登陆后跳转的页面）  
 Toast.makeText(Login.this, "登陆成功", Toast.LENGTH\_SHORT).show();  
 Intent intent = new Intent(Login.this, MainActivity.class);  
 startActivity(intent);  
 break;  
 }else {  
 Toast.makeText(Login.this, "密码错误", Toast.LENGTH\_SHORT).show();  
 break;  
 }  
 }  
 count++;  
 }  
 if (count >= object.size()){  
 Toast.makeText(Login.this,"该账号不存在",Toast.LENGTH\_SHORT).show();  
 }  
 }else {  
 Toast.makeText(Login.this,"该账号不存在",Toast.LENGTH\_SHORT).show();  
 }  
 }  
 });  
 }  
 }  
});

6.2天气信息的加载

//使用gson来解析数据  
private void parseShowData(String result) {  
 WeatherBean weatherBean = new Gson().fromJson(result, WeatherBean.class);  
 // 当天天气列表  
 WeatherBean.Newslist newslist1 = weatherBean.getNewslist().get(0);  
  
 //获取今天的温度  
 tempTv.setText(newslist1.getReal());  
 //获取城市  
 cityTv.setText(newslist1.getArea());  
 //获取天气情况  
 conditionTv.setText(newslist1.getWeather());  
 //获取日期  
 dateTv.setText(newslist1.getDate());  
 //获取风向  
 windTv.setText(newslist1.getWind());  
 //获取风力等级  
 windscTV.setText(newslist1.getWindsc());  
  
 //获取穿衣指数  
 cloth = newslist1.getTips();  
 //获取紫外线强度  
 uv = newslist1.getUv\_index();  
 //获取最高温度  
 htemp = newslist1.getHighest();  
 //获取最低温度  
 ltemp = newslist1.getLowest();  
 //获取湿度  
 cold = newslist1.getHumidity();  
  
 //获取未来六天的天气情况  
 List<WeatherBean.Newslist> newslist2 = weatherBean.getNewslist();  
 // i=0是当天的  
 for (int i = 1; i < 7; i++) {  
 // inflate就相当于将一个xml中定义的布局找出来  
 // getActivity() 用于获取fragment所绑定的activity的实例对象  
 View itemView = LayoutInflater.from(getActivity()).inflate(R.layout.item\_fragment\_center, null);  
 itemView.setLayoutParams(new LinearLayout.LayoutParams(LinearLayout.LayoutParams.MATCH\_PARENT,LinearLayout.LayoutParams.WRAP\_CONTENT));  
  
 futureLayout.addView(itemView);  
  
 // 初始化控件  
 TextView idateTv = itemView.findViewById(R.id.item\_center\_tv\_date);  
 TextView iconTv = itemView.findViewById(R.id.item\_center\_tv\_con);  
 TextView itempTv = itemView.findViewById(R.id.item\_center\_tv\_temp);  
  
 idateTv.setText(newslist2.get(i).getDate());  
 iconTv.setText(newslist2.get(i).getWeather());  
 itempTv.setText(newslist2.get(i).getReal());  
 }  
}

6.3用户找回/修改密码

// 修改密码  
ModifyButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 //获取输入的账号(Account)、密码(Password)、验证码(Code)  
 final String Account = AccountText.getText().toString().trim();  
 final String Password = PasswordText.getText().toString().trim();  
  
 String Code = SMS\_Code.getText().toString().trim();  
  
 if (TextUtils.isEmpty(Account)){  
 Toast.makeText(FindPassword.this,"请填写账号",Toast.LENGTH\_SHORT).show();  
 }else if (TextUtils.isEmpty(Code)){  
 Toast.makeText(FindPassword.this,"请填写验证码",Toast.LENGTH\_SHORT).show();  
 }else if (TextUtils.isEmpty(Password)){  
 Toast.makeText(FindPassword.this,"请填写密码",Toast.LENGTH\_SHORT).show();  
 }else if (Password.length()<6){  
 Toast.makeText(FindPassword.this,"密码不得少于6位数",Toast.LENGTH\_SHORT).show();  
 }else if (Password.length()>16){  
 Toast.makeText(FindPassword.this,"密码不得多于16位数",Toast.LENGTH\_SHORT).show();  
 }else if (!Check.PasswordCheck(Password)){  
 Toast.makeText(FindPassword.this,"密码最少包含3个字母",Toast.LENGTH\_SHORT).show();  
 }else {  
 //发送验证码  
 BmobSMS.verifySmsCode(Account, Code, new UpdateListener() {  
 @Override  
 public void done(BmobException e) {  
 if (e == null) {  
 BmobQuery<User\_Table> bmobQuery = new BmobQuery<>();  
 bmobQuery.findObjects(new FindListener<User\_Table>() {  
 @Override  
 public void done(List<User\_Table> object, BmobException e) {  
 if (e == null) {  
 for (User\_Table user\_table : object) {  
 if (user\_table.getAccount().equals(Account)){  
 //Bmob云端数据更新  
 User\_Table user = new User\_Table();  
 user.setPassword(Password);  
 user.update(user\_table.getObjectId(), new UpdateListener() {  
 @Override  
 public void done(BmobException e) {  
 if (e == null) {  
 Toast.makeText(FindPassword.this,"密码修改成功",Toast.LENGTH\_SHORT).show();  
 }else {  
 Toast.makeText(FindPassword.this,"修改失败",Toast.LENGTH\_LONG).show();  
 }  
 }  
 });  
 break;  
 }  
 }  
 }else {  
 Toast.makeText(FindPassword.this,"该账号不存在",Toast.LENGTH\_LONG).show();  
 }  
 }  
 });  
 finish();  
 }else {  
 Toast.makeText(FindPassword.this,"验证码错误",Toast.LENGTH\_SHORT).show();  
 }  
 }  
 });  
 }  
 }  
});

6.4发送验证码

// 发送验证码，account为输入的手机号，""为默认短信模板  
private void SendSMS(String account){  
 //发送短信验证码  
 BmobSMS.requestSMSCode(account, "", new QueryListener<Integer>() {  
 @Override  
 public void done(Integer integer, BmobException e) {  
 if (e == null) {  
 Toast.makeText(FindPassword.this,"验证码已发送",Toast.LENGTH\_LONG).show();  
 }else {  
 Toast.makeText(FindPassword.this,"短信发送失败",Toast.LENGTH\_LONG).show();  
 }  
 }  
 });  
  
 //设置按钮60s等待点击  
 CountDownTimer timer = new CountDownTimer(60000,1000) {  
  
 // 计时进行时的操作，显示倒计时，同时设置按钮不可点击  
 @Override  
 public void onTick(long millisUntilFinished) {  
 GetCode.setEnabled(false);  
 GetCode.setText("重新获取("+millisUntilFinished/1000+"s)");  
 }  
  
 // 计时完成时的操作，刷新原文本，同时设置按钮可以点击  
 @Override  
 public void onFinish() {  
 GetCode.setEnabled(true);  
 GetCode.setText("获取验证码");  
 }  
  
 }.start();  
}

#### **6.5主页天气Adapter** // 适配主页的天气 //FragmentStatePagerAdapter会完全销毁滑动过去的item，当需要初始化的时候，会重新初始化页面。 public class CityFragmentPagerAdapter extends FragmentStatePagerAdapter { List<Fragment>fragmentList; public CityFragmentPagerAdapter(FragmentManager fm, List<Fragment>fragmentList) { super(fm); this.fragmentList = fragmentList; } @NonNull @Override // 返回fragmentList对象 public Fragment getItem(int position) { return fragmentList.get(position); } // 返回fragmentList对象个数 @Override public int getCount() { return fragmentList.size(); } // 下面是删除城市信息时重写的方法,确认到底删除的哪个 int childCount = 0; //用于表示ViewPager包含的页数 // 当ViewPager的页数发生改变时，必须要重写两个函数 // notifyDataSetChanged方法通过一个外部的方法控制如果适配器的内容改变时需要强制调用getView来刷新每个Item的内容,可以实现动态的刷新列表的功能。 @Override public void notifyDataSetChanged() { this.childCount = getCount(); super.notifyDataSetChanged(); } // ViewPager刷新fragment @Override public int getItemPosition(@NonNull Object object) { Log.i("getItemPosition: ", String.valueOf(childCount)); if (childCount>0) { childCount--; return POSITION\_NONE; } return super.getItemPosition(object); } }

#### **6.6管理城市界面****Adapter** public class CityManagerAdapter extends BaseAdapter { Context context; List<DatabaseBean> mDatas; public CityManagerAdapter(Context context, List<DatabaseBean> mDatas) { this.context = context; this.mDatas = mDatas; } @Override public int getCount() { return mDatas.size(); } @Override public Object getItem(int position) { return mDatas.get(position); } @Override public long getItemId(int position) { return position; } @SuppressLint({"ViewHolder", "SetTextI18n"}) @Override public View getView(int position, View converView, ViewGroup viewGroup) { TextView cityTv,conTv,realTv,windTv; converView = LayoutInflater.from(context).inflate(R.layout.item\_city\_manager,null); DatabaseBean bean = mDatas.get(position); // 获取今日天气情况 WeatherBean weatherBean = new Gson().fromJson(bean.getContent(), WeatherBean.class); WeatherBean.Newslist dataBean = weatherBean.getNewslist().get(0); cityTv = converView.findViewById(R.id.item\_city\_tv\_city); conTv = converView.findViewById(R.id.item\_city\_tv\_condition); realTv = converView.findViewById(R.id.item\_city\_tv\_temp); windTv = converView.findViewById(R.id.item\_city\_wind); cityTv.setText(dataBean.getArea()); realTv.setText(dataBean.getReal()); conTv.setText("天气："+dataBean.getWeather()); windTv.setText(dataBean.getWind()); return converView; } }

6.7删除城市Adapter

public class DeleteCityAdapter extends BaseAdapter {  
  
 Context context;  
 List<String> mDatas;  
 List<String>deleteCitys;  
  
 public DeleteCityAdapter(Context context, List<String> mDatas, List<String>deleteCitys) {  
 this.context = context;  
 this.mDatas = mDatas;  
 this.deleteCitys = deleteCitys;  
 }  
  
 @Override  
 public int getCount() {  
 return mDatas.size();  
 }  
  
 @Override  
 public Object getItem(int position) {  
 return mDatas.get(position);  
 }  
  
 @Override  
 public long getItemId(int position) {  
 return position;  
 }  
  
 @SuppressLint("ViewHolder")  
 @Override  
 public View getView(int position, View convertView, ViewGroup parent) {  
 TextView tv;  
 ImageView iv;  
 convertView = LayoutInflater.from(context).inflate(R.layout.item\_deletecity,null);  
  
 tv = convertView.findViewById(R.id.item\_delete\_tv);  
 iv = convertView.findViewById(R.id.item\_delete\_iv);  
  
 final String city = mDatas.get(position);  
 tv.setText(city);  
  
 iv.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 mDatas.remove(city);  
 deleteCitys.add(city);  
 notifyDataSetChanged(); //删除了要提示适配器更新  
 }  
 });  
  
 return convertView;  
 }  
  
}