目 录

第一章 绪论 1

1.1 系统开发意义 1

1.2 研究内容 1

第二章 需求分析与设计 2

2.1 系统需求分析 2

2.2 系统功能分析 2

2.3 系统功能 2

2.4 系统流程分析 3

2.5 系统非功能性需求分析 3

第三章 系统的详细设计与实现 4

3.1 客户端实现 4

3.1.1 登录模块客户端页面 4

3.1.2 注册页面 5

3.2 定位模式 6

3.2.1 GPS定位 6

3.2.2网络定位模式 7

3.3 多城市选择页面 7

3.4 天气显示页面 8

3.5. 手势滑动 9

3.6 天气模糊背景 9

3.7 网络接口模块 10

3.8 小结 11

第四章 系统测试 12

4.1 系统测试 12

4.2 小结 13

第五章 总结与展望 14

参考文献 15

**摘 要：**这个系统是基于物联网移动终端的应用，能够随时随地进行查询，不受地域以及时间的限制，能够实时更新，将最新天气信息传递给用户，让用户感知天气更加方便，不需要每天定时观看新闻播报，用户可以通过该软件及时作出应变，能够充分地减少天气突变带来的损失。因此，我开发的这款基于Android的天气预报软件设计是至关重要的，主要采用JSP技术，整个系统操作方便，易于维护、灵活实用。

**关键词：**天气预报；移动终端；Android；实时更新

# 第一章 绪论

## 1.1 系统开发意义

该系统的开发能够让气象信息覆盖面有效扩大，用户可以不分时间地点，快速获取气象信息，做好预防应对，可以有效的控制气象灾害损失。气象公共服务职能得到进一步强化，使得公共服务体系得到进一步完善，促进气象事业的可持续发展。同时气象信息的实时更新也是气象灾害控制的一个重要难题，如果能够实施气象信息的实时更新，将会对工业，农业等等方面有着巨大的帮助。本系统的预计希望能开发一款界面简洁大方，反应迅速灵敏，适用于广大用户的天气预报软件。

## 1.2 研究内容

该系统主要研究的是天气预报系统，通过Android平台，获取网络服务的支持，以HTTP的方式接入气象网接口，获取实时天气信息。同时能够根据用户的位置，获取当地的天气信息，实现多个城市天气信息的实时监控和对城市的增加减少。支持多用户同时使用一个客户端。

# 第二章 需求分析与设计

## 2.1 系统需求分析

（1）系统能够通过手机精准定位全国各个城市未来几天的天气情况。

（2）系统具有安全性，独立性，可靠性。

（3）系统UI设计优美，符合用户操作习惯，且在一定程度上节省流量。

## 2.2 系统功能分析

1. 用户注册/登录：实现用户的注册和登录，只有当密码和账号相对应，且输入一致时才能正确登录使用该系统。

2. GPS定位：调用安卓SDK的定位中的LocationClient和BDLocation等，以及BDLocationListener等接口，通过GPS和网絡来返回百度坐标系的定位结果[7]，在用户联网情况下如WIFI和移动数据开启，可以根据百度地图定位当前城市信息进行实时信息更新。

3. 多城市选择：一个是热门城市模块部分，另一个是所有城市模块部分，国内主要的大城市是在热门城市模块中体现的，方便用户选择，用户可以通过输入的方式，来选择其他目标城市。选择完毕后可以通过确定按钮来提交设置。同一用户可以选择多个城市，并获取相应的天气情况。

4. 天气显示：可以联网，访问中国气象网，添加接口，获取天气预报、温湿度、空气状况及其他天气指数。

5. 界面设置：采用和当时天气状况相对应的图片作为主界面，未登录时默认采用阳光明媚的图片。其他信息采用简洁明了的方式呈现给用户。

## 2.3 系统功能

通过需求分析，确定系统的主要功能如下，功能框架如图3-1所示；

实时天气信息:用户可以实时的，对所选择的城市进行具体信息查询，包括目标城市的温湿度、风力指数和穿衣指数以及出行方式等;

未来三天预报:用户可以查看到未来三天的天气情况；

更换城市:用户对城市进行切换，选择想要查询的目标城市；

GPS定位:可以获得当前用户所在的城市。

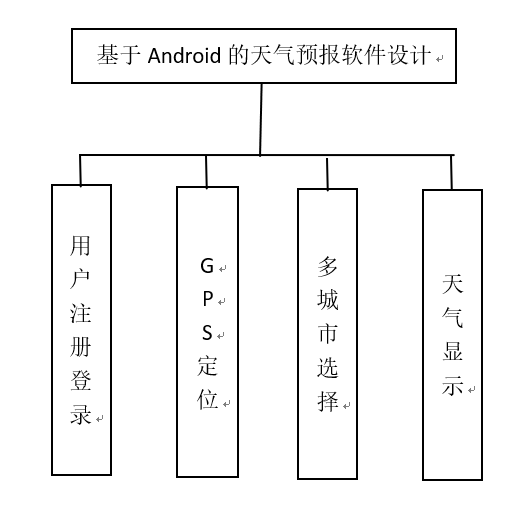
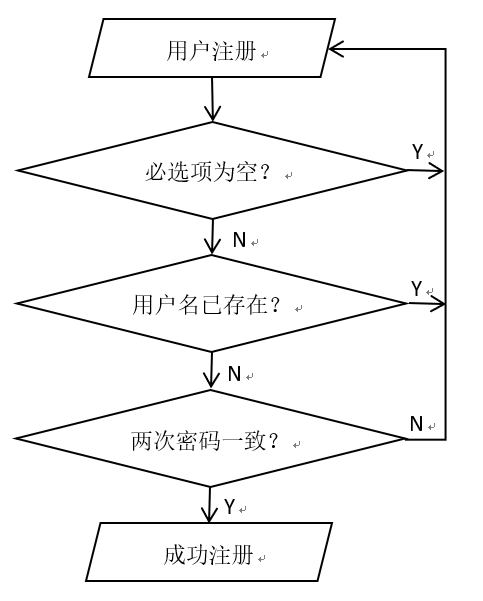
 

图2.1 系统功能结构图 图2.2 用户注册流程

## 2.4 系统流程分析

该系统的流程比较清晰，首先点击注册按钮，填入用户名密码，以及确认密码三个选项，若密码一致，则注册成功，若不一致，则重新输入，注册具体流程如图3-2所示。

## 2.5 系统非功能性需求分析

1. 系统性能分析

对于系统性能的分析分为两部分，吞吐率以及响应率。通过检测某一指定用户组在登录系统后的数据，对数据进行监测与分析，确定其是否满足要求。因此，系统测试的重要性不言而喻。监测的数据主要有两种指标，其一是吞吐量，其二是响应时间。吞吐率以及响应时间彼此相互对应。

2. 系统安全

系统的安全性是指在使用系统期间用户的数据使用是否安全，系统操作的数据文件是否安全，以及系统的速度和效率是否稳定。

3. 系统稳定

系统的稳定性主要是指系统程序在运行过程中需要保证数据文件的稳定性和安全性。要备份的数据库和目前的数据库在系统运行的过程中需要实现同步运行。数据库与运行数据库之间的同步[9]确保了数据文件的安全稳定运行。

# 第三章 系统的详细设计与实现

## 3.1 客户端实现

### 3.1.1 登录模块客户端页面

在登录模块界面的设置上，选择界面中最常见的方式之一的对话框模式。用list对象存储用户输入的账户和密码。主要代码如下，界面如图3。

/\*\*

\* 登录

\*/

**public** **void** login(View view) {

currentUsername = usernameEditText.getText().toString().trim();

currentPassword = passwordEditText.getText().toString().trim();

**if** (TextUtils.*isEmpty*(currentUsername)) {

Toast.*makeText*(**this**, "请输入用户名", Toast.*LENGTH\_SHORT*).show();

**return**;

}

**if** (TextUtils.*isEmpty*(currentPassword)) {

Toast.*makeText*(**this**, "请输入密码", Toast.*LENGTH\_SHORT*).show();

**return**;

}

SharedPreferences sharedPreferences = getSharedPreferences("test",

Activity.*MODE\_PRIVATE*);

String name = sharedPreferences.getString("name", "").trim();

String word = sharedPreferences.getString("word", "").trim();

**if** (name.equals(currentUsername) && word.equals(currentPassword)) {

startActivity(**new** Intent(**this**, MainActivity.**class**));

finish();

} **else** {

Toast.*makeText*(**this**, "账号或密码输入错误", Toast.*LENGTH\_SHORT*).show();

}

}

图3.1用户登录页面 图3.2 注册页面

### 3.1.2 注册页面

用一个list对象，将用户所输入的信息，也就是账户密码进行，主要代码如下，实现效果见图4。

/\*\*

\* 注册

\*/

**public** **void** register(View view) {

**final** String username = userNameEditText.getText().toString().trim();

**final** String pwd = passwordEditText.getText().toString().trim();

String confirm\_pwd = confirmPwdEditText.getText().toString().trim();

**if** (TextUtils.*isEmpty*(username)) {

Toast.*makeText*(**this**, "用户名不能为空", Toast.*LENGTH\_SHORT*).show();

userNameEditText.requestFocus();

**return**;

} **else** **if** (TextUtils.*isEmpty*(pwd)) {

Toast.*makeText*(**this**, "密码不能为空", Toast.*LENGTH\_SHORT*).show();

passwordEditText.requestFocus();

**return**;

} **else** **if** (TextUtils.*isEmpty*(confirm\_pwd)) {

Toast.*makeText*(**this**, "请确认密码", Toast.*LENGTH\_SHORT*).show();

confirmPwdEditText.requestFocus();

**return**;

} **else** **if** (!pwd.equals(confirm\_pwd)) {

Toast.*makeText*(**this**, "密码输入不一样", Toast.*LENGTH\_SHORT*).show();

**return**;

}

}

## 3.2 定位模式

### 3.2.1 GPS定位

用户可以通过引用百度地图定位，获得当前城市信息。在此之间还需要获取GPS定位的权限，其代码在AndroidManifest.xml中。GPS定位主要代码如下，实现效果见图5。

/\*\*

\* 实现定位回调监听

\*/

BDLocationListener mLocationListener = **new** BDLocationListener() {

@Override

**public** **void** onReceiveLocation(BDLocation location) {

**if** (location == **null** || location.getLocType() != 161

|| TextUtils.*isEmpty*(location.getCity())) {

mTryCount++;

**if** (mTryCount >= *MAX\_TRY\_COUNT*) {

mListener.failed();

stopLocation();

}

**return**;

}

String city = location.getCity().replace("市", "");

mListener.succeed(city);

stopLocation();// 停止定位

}

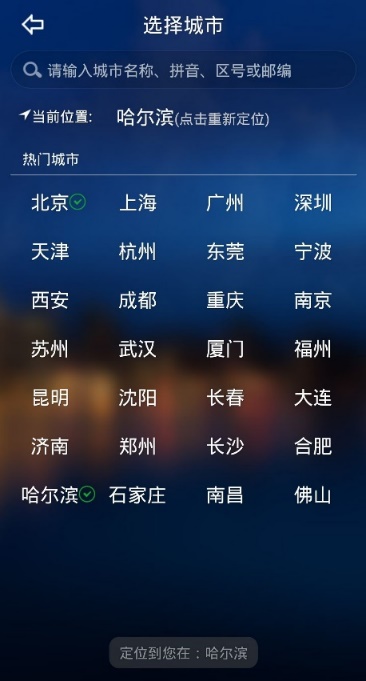
 

图3.3 GPS定位页面 图3.4 多城市管理界面

### 3.2.2网络定位模式

用户可以通过WiFi和流量的方式，获取当前城市的信息，在此之前还需要获取WiFi通信的权限，其代码在AndroidManifest.xml中。网络定位主要代码如下。

// Wifi

State state = connManager.getNetworkInfo(ConnectivityManager.*TYPE\_WIFI*)

.getState();

**if** (state == State.*CONNECTED* || state == State.*CONNECTING*) {

**return** *NETWORN\_WIFI*;

}

// 3G

state = connManager.getNetworkInfo(ConnectivityManager.*TYPE\_MOBILE*)

.getState();

**if** (state == State.*CONNECTED* || state == State.*CONNECTING*) {

**return** *NETWORN\_MOBILE*;

}

**return** *NETWORN\_NONE*;

}

## 3.3 多城市选择页面

用户可以通过app可以选择多个城市，并显示相应的天气情况。主要代码如下，实现效果如图3.4。

**private** **void** changeSortIndex() {

**for** (**int** i = 0; i < mTmpCitys.size(); i++) {

ContentValues contentValues = **new** ContentValues();

contentValues.put(CityConstants.*ORDER\_INDEX*, i);

String postID = mTmpCitys.get(i).getPostID();

mContentResolver.update(CityProvider.*TMPCITY\_CONTENT\_URI*,

contentValues, CityConstants.*POST\_ID* + "=?",

**new** String[] { postID });// 更新位置

}

}

用户通过城市管理来实现对城市的编辑，主要代码如下：

**private** **void** deleteCityFromTable(**int** position) {

City city = mAdapter.getItem(position);

// 从临时城市表中删除

mContentResolver

.delete(CityProvider.*TMPCITY\_CONTENT\_URI*, CityConstants.*POST\_ID*

+ "=?", **new** String[] { city.getPostID() });

updateUI(**false**);

**if** (mTmpCitys.isEmpty())// 如果全部被删除完了，更新一下编辑状态

changeEditMode();

changeSortIndex();

}

## 3.4 天气显示页面

此页面为用户功能，可以显示天气预报、温湿度、空气状况及其他天气指数。主要代码如下，实现效果如图7。

**public** WeatherInfo(RealTime realTime, Forecast forecast, AQI aqi,

Index index, Alerts alerts) {

**super**();

**this**.realTime = realTime;

**this**.forecast = forecast;

**this**.aqi = aqi;

**this**.index = index;

**this**.alerts = alerts;

}

侧边栏的实现在net.simonvt.menudrawer和net.simonvt.menudrawer.compat两个包里，这里不多赘述。



图3.5 天气显示页面 图3.6 侧边栏界面

## 3.5. 手势滑动

该功能在软件设计中是必不可少的，实现了美化APP的效果。本设计将其应用在从主页面到侧边栏的滑动效果。

**private** Handler mHandler = **new** Handler() {

**public** **void** handleMessage(android.os.Message msg) {

**switch** (msg.what) {

**case** *ACTIVITY\_TIMEOUT\_GOTO\_NEXT*:

Intent intent = **new** Intent();

intent.setClass(SplashActivity.**this**, LoginActivity.**class**);

startActivity(intent);

SplashActivity.**this**.finish();

**break**;

**default**:

**break**;

}

}

};

## 3.6 天气模糊背景

在天气预报的APP中，背景是展示的关键。不同的时间段和天气情况下，背景需要进行很多变化。并且在浏览时由于背景自带的色彩在一定程度上影响了用户的体验，所以本设计在用户进行浏览时对背景进行了虚化，提供模糊背景。主要代码如下：

/\*\*

\* 获取天气模糊背景

\*/

**public** **static** **int** getWeatherBlurBg(**int** type) {

**if** (*isNight*(System.*currentTimeMillis*()))

**switch** (type) {

**case** Constants.*SUNNY*:

**return** R.drawable.*bg\_fine\_night\_blur*;

**case** Constants.*CLOUDY*:

**return** R.drawable.*bg\_cloudy\_night\_blur*;

**case** Constants.*HEAVY\_RAIN*:

**case** Constants.*LIGHT\_RAIN*:

**case** Constants.*MODERATE\_RAIN*:

**case** Constants.*SHOWER*:

**case** Constants.*ICE\_RAIN*:

**return** R.drawable.*bg\_rain\_blur*;

**case** Constants.*STORM*:

**return** R.drawable.*bg\_rain\_blur*;

**case** Constants.*SNOWSTORM*:

**case** Constants.*LIGHT\_SNOW*:

**case** Constants.*MODERATE\_SNOW*:

**case** Constants.*HEAVY\_SNOW*:

**case** Constants.*SNOW\_SHOWER*:

**return** R.drawable.*bg\_snow\_night\_blur*;

**default**:

**break**;

}

## 3.7 网络接口模块

通过HTTP请求发送方式，得到城市天气相关的数据，数据成功得到后，可得到最新城市的天气信息，主要代码如下：

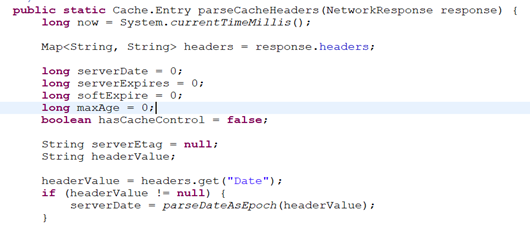


图3.7 网络接口模块

## 3.8 小结

预设的功能经努力基本能够实现，代码bug也修改结束，经过初步运行，软件能够正常运行，且各个界面显示正常，系统的稳定性和兼容性，还需进行后续的测试验证。

# 第四章 系统测试

系统测试十分关键，在项目开发中扮演着重要的角色。系统测试可以使得软件质量得到保障，作为软件工程的关键部分，随着软件越来越复杂，有着越来越大的规模，依托于系统测试，这样软件质量就可以得到很好的保障，改善产品的问题，使得软件更加完美。

## 4.1 系统整体测试

1. “更换城市”测试

首先点击主界面左上角的菜单按钮，修改当前城市，进行增加与删除，系统当前的城市发生变更，且更新了城市数据。如图8、图9所示。

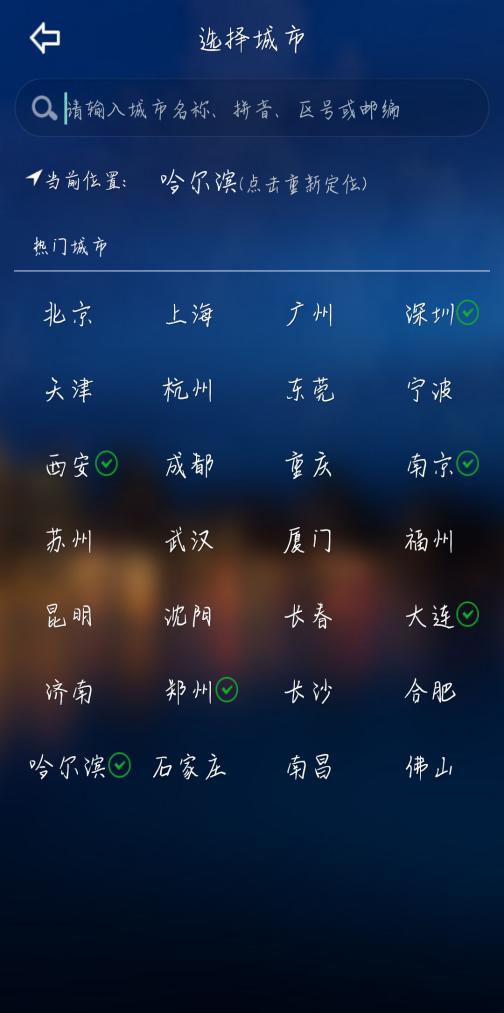
 

图4.1 城市添加界面 图4.2 选择城市界面

(2)“自动定位”测试

点击按钮，可以获得用户所在的城市。

(3)异常处理

当用户没有联网，或者模拟器运行时间太长时，下拉界面刷新，界面显示数据正在刷新，但得不到实时数据，此时系统会进行异常处理。如图10、图11。

图4.3 未联网时空气质量情况 图4.4 联网后空气质量情况

## 4.2 小结

本次测试我们分为三个方面进行测试，分别是主页面、更换城市界面与网络接口三个部分，具体步骤如下：首先进入主页面，在主页面左上角找到菜单按钮，点击进入菜单页面寻找更改城市按钮，将城市修改为其他城市——南京，点击确定按钮，发现城市界面已经更换为南京，证明主页面及更换城市界面模块无误，观察到显示的天气数据发生变化，与南京实时天气进行对比，验证数据正确，得出结论“本系统主界面、更换城市界面和网络接口三个模块能够正常工作”。

# 第五章 总结与展望

本系统主要实现了基于Android平台的天气预报的软件设计与实现过程，提出了Android用户页面设计、获取并解析城市列表数据的一种方法，给出了在用户界面上呈现数据的原理与设计过程，最后通过模拟器进行了应用程序的调试。

本系统还存在各种方面的不足，在气象预报的功能上，预计在后续研究中希望能够与各个地区的气象站进行数据共享，完善系统中的数据，根据人们穿衣的习惯与天气的分析相结合，智能推荐衣服以及出行方式，与其他的软件相结合，开发出功能日益完善的软件体系。

# 参考文献

[1] 侯培文. 软件工程中软件测试技术的研究[J]. 电子技术与软件工程, 2016(8), 72-72.

[2] 王靖. 计算机操作系统运行与维护研究[J].无线互联科技, 2017, (22).

[3] RuiZHANG, HuiMA, YaoLU, YangLI. Provably secure cloud storage for mobile networks with lesscomputation and smaller overhead[J]. Science China Information Sciences, 2017, 60, (12).

[4] 罗浩，王丹，肖络元，陈渝. Android多窗口系统的研究、设计与实现[J]. 计算机科学，2017, 44 (11).

[5] 李涵，耿庆仁. Android系统下校园e超市软件设计[J].中国科技信息, 2017,(22).

[6] 晏杰. 基于Android平台网上预约挂号管理系统的设计与实现[J]. 榆林学院学报，2017(06), 27.

[7] 顾学海，王聪颖，胡牧，蒋厚明，曹海涛. Android移动应用的三种安全解决方法探究[J]. 计算机系统应用, 2017, (11).

[8] 刘志成，宁云智编著. 数据库系统原理与应用 [M]. 机械工业出版, 2010, 03.

[9] 明日科技著. JSP项目开发实战入门. 吉林大学出版社, 2017.

[10] 贾志城,王云编. JSP程序设计慕课版. 人民邮电出版社, 2016.

[11] (美)Cay S. Horstmann. Java核心技术-卷Ⅱ, Volume II-高级特性, 人民邮电出版社, 2017.

[12] 张瑶，丁胜，乐东东，张潇. 电脑知识与技术，2016, 12(05), 100-102.

**Design of Weather Forecasting Software**

**Based on Android**

**Abstract:** Since the first industrial revolution, human science and technology has entered an era of rapid development, such as transportation from the most primitive livestock, a little into the era of steam engine, and eventually evolved into today's electrical era, communication development is the same. Today, with the rapid development of science and technology, people have more and more diversified ways of obtaining information, because people now have more and more people. Mobile phones are becoming more and more dependent, so my graduation project chose the weather forecast system. This system is based on the application of mobile terminals on the Internet of Things. It can query anytime, anywhere, without geographical and time constraints. It can update in real time and transmit the latest weather information to users, so that users can perceive the weather more conveniently and do not need to watch news broadcasts regularly every day. Users can make timely response through the software, which can fully reduce the loss caused by abrupt weather changes. Therefore, the design of this weather forecast software based on Android is very important. It mainly uses JSP technology. The whole system is easy to operate, easy to maintain, flexible and practical.

**Keywords:** weather report; mobile terminal; Android; JSP; Real time update