

# Lab9 实验报告

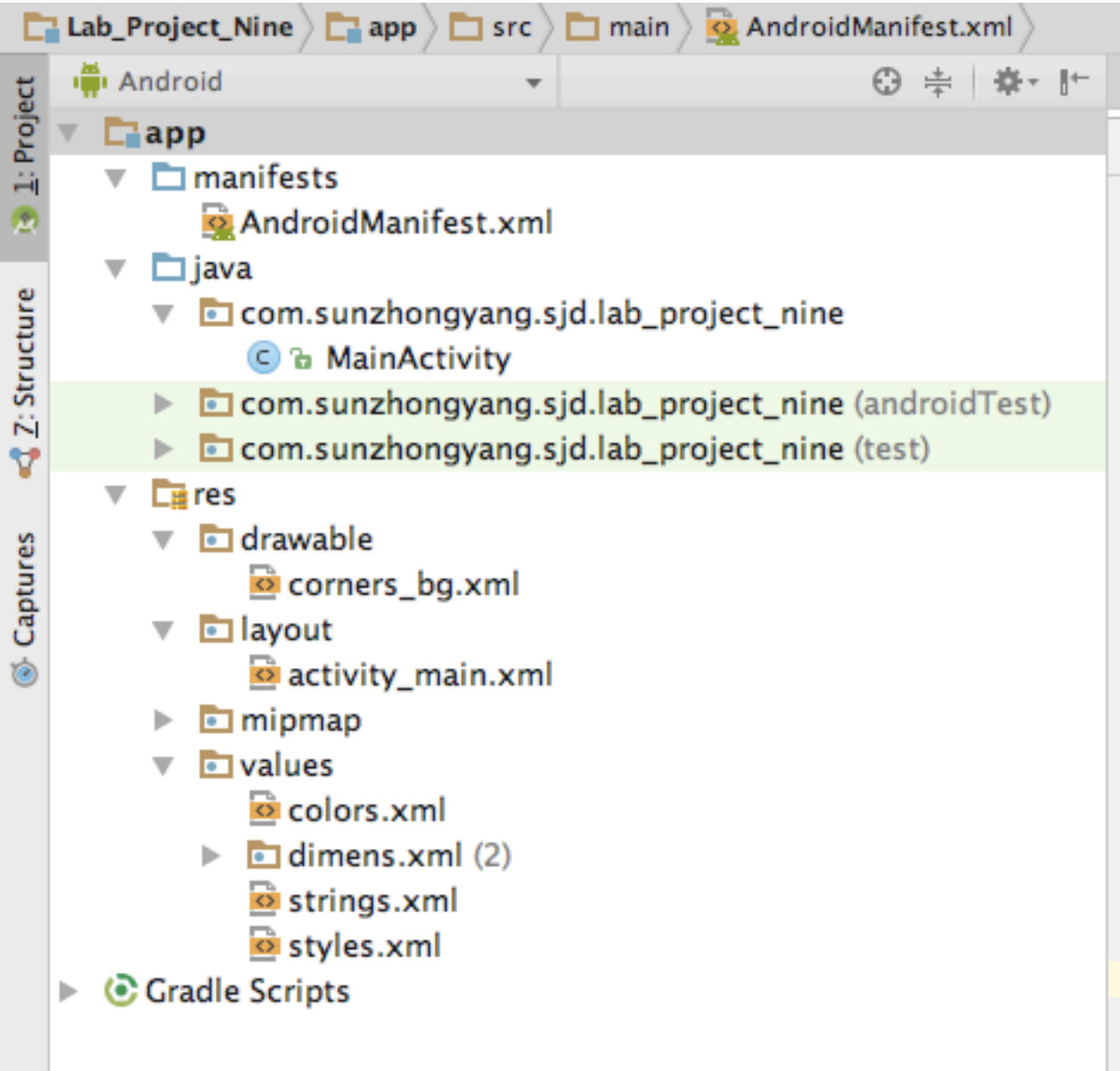
13331233 孙中阳

## 参考资料

《第一行代码 Android》 郭霖 著  
《Lab9实验文档》

## 实验步骤

本次试验代码量较上一次实验有所提升，难度却有所下降。主要需要熟悉 网络编程及 `XmlPullParser` 的使用并使用 `handler` 经线程控制UI等  
图为项目结构

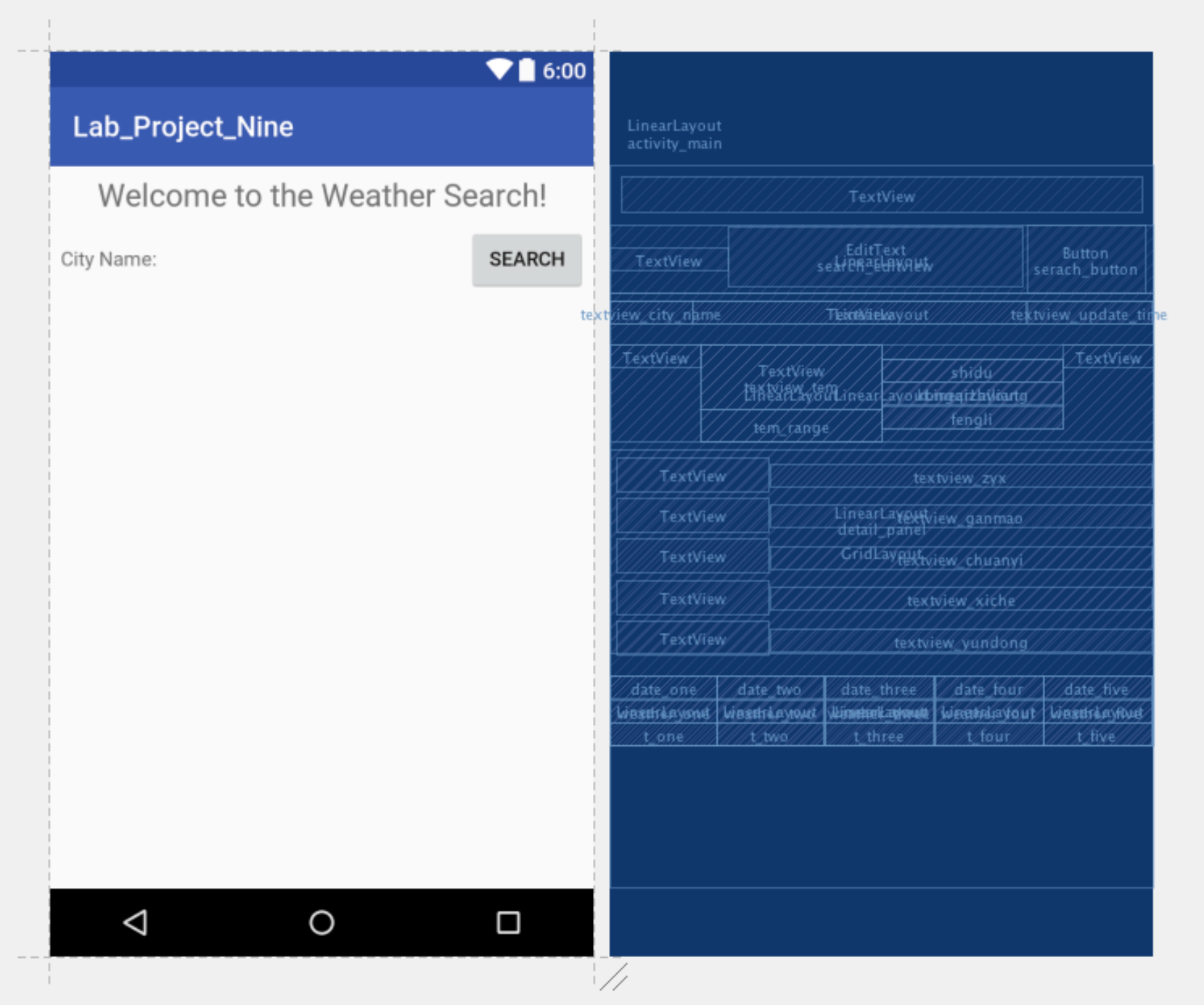


从实验文档看，需要实现的目标主要有

- 1、使用 `HttpURLConnection` 访问 `WebService`
- 2、使用 `XmlPullParser` 解析 xml 文档数据

3、使用多线程以及 `Handler` 更新 UI

图为主界面布局，布局为历次作业中最复杂的一次



接下来考虑实现  
本次开发主要涉及三个方面：

1.通过网络获得天气信息

首先需要获得连接网络及查看网络连接状态的权限，在 `AndroidManifest.xml` 中增添以下内容

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

查看网络状态，如果没有连接则弹窗提示

//判断当前是否已连接到网络

```
public boolean isConnected(Activity activity)
{
    //获取网络连接状态
    Context context = MainActivity.this.getApplicationContext();
    ConnectivityManager connectivityManager = (ConnectivityManager)context.getSystemService(Context.CONNECTIVITY_SERVICE);

    NetworkInfo[] networkInfo = connectivityManager.getAllNetworkInfo();

    for (int i = 0; i < networkInfo.length; i++)
    {
        if (networkInfo[i].getState() == NetworkInfo.State.CONNECTED)
        {
            return true;
        }
    }
    return false;
}
```

然后在新建的线程中执行网络访问，url 之前已填写

```

URLConnection connection = null;
try
{
    connection = (URLConnection) ((new URL(url.toString())).openConnection());
    connection.setRequestMethod("POST");
    connection.setReadTimeout(8000);
    connection.setConnectTimeout(8000);

    //向服务器发送信息
    DataOutputStream out = new DataOutputStream(connection.getOutputStream());
    String request = URLEncoder.encode(city_name, "utf-8");

    out.writeBytes("theCityCode=" + request + "&theUserID=");

    //读取服务器返回的信息
    InputStream in = connection.getInputStream();
    BufferedReader reader = new BufferedReader(new InputStreamReader(in));
    StringBuilder response = new StringBuilder();

    String line;

    while((line = reader.readLine()) != null)
    {
        response.append(line);
    }

    //向handler对象发送消息以传递数据
    Message message = new Message();
    message.what = 0;
    message.obj = parseXMLWithPull(response.toString());
    handler.sendMessage(message);
}
catch (Exception e)
{
    e.printStackTrace();
}
//断开连接
finally
{
    if(connection != null)
    {
        connection.disconnect();
    }
}

```

这里使用函数 `parseXMLWithPull()` 将收到的来自服务器的回复存储在一个 `ArrayList` 中

最后使用 `handler` 向线程外发送收到的报文转换成的 `list`

## 2.通过 `parseXMLWithPull()` 解析 xml 文档数据

解析 xml 文档数据，必须要使用到 `XmlPullParser`，其具体使用方法为

```
//将服务器返回的信息转换为字符串链表
public ArrayList<String> parseXMLWithPull(String response)
{
    ArrayList<String> list = new ArrayList<String>();

    //读取xml文本
    try
    {
        XmlPullParserFactory factory = XmlPullParserFactory.newInstance();
        XmlPullParser parser = factory.newPullParser();
        parser.setInput(new StringReader(response));

        //在文件截止前逐行读取
        int eventType = parser.getEventType();
        while(eventType != XmlPullParser.END_DOCUMENT)
        {
            switch(eventType)
            {
                case XmlPullParser.START_TAG:
                    if(parser.getName().equals("string"))
                    {
                        String str = parser.nextText();
                        list.add(str);
                    }
                    break;
                case XmlPullParser.END_TAG:
                    break;
                default:
                    break;
            }
            eventType = parser.next();
        }
    }
    catch(Exception e)
    {
        e.printStackTrace();
    }

    return list;
}
```

简单的说就是逐标签读数据，直到文件结尾，独到的内容保存在 `ArrayList` 中

## 3.通过 `handler` 更新UI

目前已有通过 `handler` 发送来的 `ArrayList`，但首先要知道如何构建一个 `handler`

详细方法如下

```
//建立一个handler从线程接收数据,并根据情况决定UI
private Handler handler = new Handler()
{
    public void handleMessage(Message message)
    {
        switch (message.what)
        {
            //根据服务器返回消息类型决定对应的操作
            case 0:
                ArrayList<String> list = (ArrayList<String>) message.obj;
                if(list.get(0).equals("发现错误: 免费用户24小时内访问超过规定数量。http://www.webxml.com.cn/"))
                {
                    Toast.makeText(MainActivity.this, "免费用户24小时内访问超过规定数量50次", Toast.LENGTH_SHORT).show();
                }
                else if(list.get(0).equals("查询结果为空。http://www.webxml.com.cn/"))
                {
                    Toast.makeText(MainActivity.this, "当前城市不存在,请重新输入", Toast.LENGTH_SHORT).show();
                }
                else if(list.get(0).equals("发现错误:免费用户不能使用高速访问。http://www.webxml.com.cn/"))
                {
                    Toast.makeText(MainActivity.this, "您的点击速度过快,二次查询间隔<600ms", Toast.LENGTH_SHORT).show();
                }
                else
                {
                    //如果一切正常,则更新UI
                    updateUI(list);
                }
                break;
            default:
                break;
        }
    }
};
```

这里首先判断 `message` 携带的消息,如果正确则根据其数据判断需要执行的操作,如果包含了正确的天气信息,则通过 `updateUI()` 函数更新界面

```
//更新UI
public void updateUI(ArrayList<String> list)
{
    //获取各个控件并且根据list的内容初始化其内容
    LinearLayout detail_panel = (LinearLayout) findViewById(R.id.detail_pane
```

```
1);
detail_panel.setVisibility(Layout.VISIBLE);

TextView textview_city_name = (TextView) findViewById(R.id.textview_city_name);
textview_city_name.setText(list.get(1));

TextView textview_update_time = (TextView) findViewById(R.id.textview_update_time);
textview_update_time.setText(list.get(3).split(" ")[1] + " 更新");

TextView textview_tem = (TextView) findViewById(R.id.textview_tem);
textview_tem.setText(list.get(4).split(": ")[2].split("; ")[0]);

TextView tem_range = (TextView) findViewById(R.id.tem_range);
tem_range.setText(list.get(8));

TextView shidu = (TextView) findViewById(R.id.shidu);
shidu.setText(list.get(4).split(": ")[4]);

TextView kongqizhiliang = (TextView) findViewById(R.id.kongqizhiliang);
kongqizhiliang.setText(list.get(5).split(": ")[2]);

TextView fengli = (TextView) findViewById(R.id.fengli);
fengli.setText(list.get(4).split(": ")[3].split("; ")[0]);

TextView textview_zyx = (TextView) findViewById(R.id.textview_zyx);
textview_zyx.setText(list.get(6).split(": ")[1].split("。 ")[0]);

TextView textview_ganmao = (TextView) findViewById(R.id.textview_ganmao);
;
textview_ganmao.setText(list.get(6).split(": ")[2].split("。 ")[0]);

TextView textview_chuanyi = (TextView) findViewById(R.id.textview_chuanyi);
textview_chuanyi.setText(list.get(6).split(": ")[3].split("。 ")[0]);

TextView textview_xiche = (TextView) findViewById(R.id.textview_xiche);
textview_xiche.setText(list.get(6).split(": ")[4].split("。 ")[0]);

TextView textview_yundong = (TextView) findViewById(R.id.textview_yundong);
textview_yundong.setText(list.get(6).split(": ")[5].split("。 ")[0]);

TextView date_one = (TextView) findViewById(R.id.date_one);
date_one.setText(list.get(7));

TextView weather_one = (TextView) findViewById(R.id.weather_one);
weather_one.setText(list.get(8));

TextView t_one = (TextView) findViewById(R.id.t_one);
```

```

t_one.setText(list.get(9));

TextView date_two = (TextView) findViewById(R.id.date_two);
date_two.setText(list.get(12));

TextView weather_two = (TextView) findViewById(R.id.weather_two);
weather_two.setText(list.get(13));

TextView t_two = (TextView) findViewById(R.id.t_two);
t_two.setText(list.get(14));

TextView date_three = (TextView) findViewById(R.id.date_three);
date_three.setText(list.get(17));

TextView weather_three = (TextView) findViewById(R.id.weather_three);
weather_three.setText(list.get(18));

TextView t_three = (TextView) findViewById(R.id.t_three);
t_three.setText(list.get(19));

TextView date_four = (TextView) findViewById(R.id.date_four);
date_four.setText(list.get(22));

TextView weather_four = (TextView) findViewById(R.id.weather_four);
weather_four.setText(list.get(23));

TextView t_four = (TextView) findViewById(R.id.t_four);
t_four.setText(list.get(24));

TextView date_five = (TextView) findViewById(R.id.date_five);
date_five.setText(list.get(27));

TextView weather_five = (TextView) findViewById(R.id.weather_five);
weather_five.setText(list.get(28));

TextView t_five = (TextView) findViewById(R.id.t_five);
t_five.setText(list.get(29));
}

```

这里看起来很复杂，其实很简单，复杂性来源于本次呈现信息较多的布局

## 实验成果截图

初始界面





# Welcome to the Weather Search!

City Name:

SEARCH

开启了飞行模式，没有网络

# Lab\_Project\_Nine

Welcome to the Weather Search!

City Name:

SEARCH

当前没有可用的网络!

正常查询

20:47

9.80K/s 9.80K/s 4G 100%

# Lab\_Project\_Nine

Welcome to the Weather Search!

City Name: 广州

SEARCH

广州

20:37:27 更新

17°C

60%

11°C/18°C

良。

北风 1级

紫外线指数 弱，辐射较弱，涂擦SPF12-15、PA+护肤品

感冒指数 较易发，天较凉，增加衣服，注意防护

穿衣指数 较冷，建议着厚外套加毛衣等服装

洗车指数 较适宜，无雨且风力较小，易保持清洁度


# 运动指数

较适宜，推荐进行室内运动

11月30日 多云	12月1日 多云	12月2日 多云	12月3日 多云	12月4日 多云
11°C/18°C	12°C/19°C	13°C/20°C	14°C/22°C	15°C/23°C
北风3-4级	无持续风向 微风	无持续风向 微风	无持续风向 微风	无持续风向 微风

城市不存在

20:47

0.00K/s   4G  100%

Lab\_Project\_Nine

Welcome to the Weather Search!

City Name: 孙中阳

SEARCH

广州

20:37:27 更新

17°C

60%  
良。

11°C/18°C

北风 1级

# 紫外线指数

22 辐射较弱 涂擦SPF12-15 PA+防晒品

## 感冒指数

较易发，天较凉，增加衣服，注意防护

## 穿衣指数

较冷，建议着厚外套加毛衣等服装

## 洗车指数

较适宜，无雨且风力较小，易保持清洁度

## 运动指数

较适宜，推荐进行室内运动

11月30日 多云	12月1日 多云	12月2日 多云	12月3日 多云	12月4日 多云
11°C/18°C	12°C/19°C	13°C/20°C	14°C/22°C	15°C/23°C
北风3-4级	无持续风向 微风	无持续风向 微风	无持续风向 微风	无持续风向 微风

当前城市不存在,请重新输入

点击速度过快

20:50

82.0K/s   4G  100%

# Lab\_Project\_Nine

# Welcome to the Weather Search!

City Name: 广州

## SEARCH

广州

20:37:27 更新

17°C

60%

良。

11°C/18°C

北风 1级

**紫外线指数** 弱，辐射较弱，涂擦SPF12-15、PA+护肤品

**感冒指数** 较易发，天较凉，增加衣服，注意防护

**穿衣指数** 较冷，建议着厚外套加毛衣等服装

**洗车指数** 较适宜，无雨且风力较小，易保持清洁度

**运动指数** 较适宜，推荐进行室内运动

11月30日 多云	12月1日 多云	12月2日 多云	12月3日 多云	12月4日 多云
11°C/18°C	12°C/19°C	13°C/20°C	14°C/22°C	15°C/23°C
北风3-4级	无持续风向 微风	无持续风向 微风	无持续风向 微风	无持续风向 微风

您的点击速度过快,二次查询间隔<600ms

我使用一个循环以一秒一次的间隔访问了数十次，最终超过规定数量并获得预期结果

20:47

5.59K/s     100%

Lab\_Project\_Nine

# Welcome to the Weather Search!

City Name: 广州

SEARCH

广州

20:37:27 更新

17°C

60%

良。

11°C/18°C

北风 1级

**紫外线指数** 弱，辐射较弱，涂擦SPF12-15、PA+护肤品

**感冒指数** 较易发，天较凉，增加衣服，注意防护

**穿衣指数** 较冷，建议着厚外套加毛衣等服装

**洗车指数** 较适宜，无雨且风力较小，易保持清洁度

**运动指数** 较适宜，推荐进行室内运动

11月30日 多云	12月1日 多云	12月2日 多云	12月3日 多云	12月4日 多云
-----------	----------	----------	----------	----------

11°C/18°C	12°C/19°C	13°C/20°C	14°C/22°C	15°C/23°C
-----------	-----------	-----------	-----------	-----------

北风3-4级	无持续风向 微风	无持续风向 微风	无持续风向 微风	无持续风向 微风
--------	-------------	-------------	-------------	-------------

免费用户24小时内访问超过规定数量50次

# 实验过程遇到的问题

---

这次并没有遇到很大的问题，新知识以外的部分可参考之前的代码完成。实现方式按照文档给出的非 `Ksoap2` 方式，有的时候网络会报错，但对正常使用没有影响

比较卡时间的是我已开始没有看到网络部分要单独放在一个进程中，故在建立 `DataOutputStream` 出现错误导致程序停止运行，后来查询网络和文档才发现问题原因

另外一个地方是服务器返回的字符串采用 `utf-8` 编码的汉字，其中的冒号和英文的冒号不一致，导致分割文档过程中出现问题，后经复制 log 中的字符到程序中替换掉英文字符，问题解决

## 思考与总结

---

本次试验深入了解了Android开发的有关知识，尤其是和网络访问以及 `XmlPullParser` 有关的内容。这次实验的内容较丰富，尤其是 UI 部分需要实现的略繁琐的工作。其中大部分内容可以通过查阅PPT或作业说明得到解决，只有少部分需要百度。

原理部分更多的参考了上课的课件。Android较UWP应用范围更广，网上的资料也更多，问题也解决的比较顺利。总的来说比较有收获