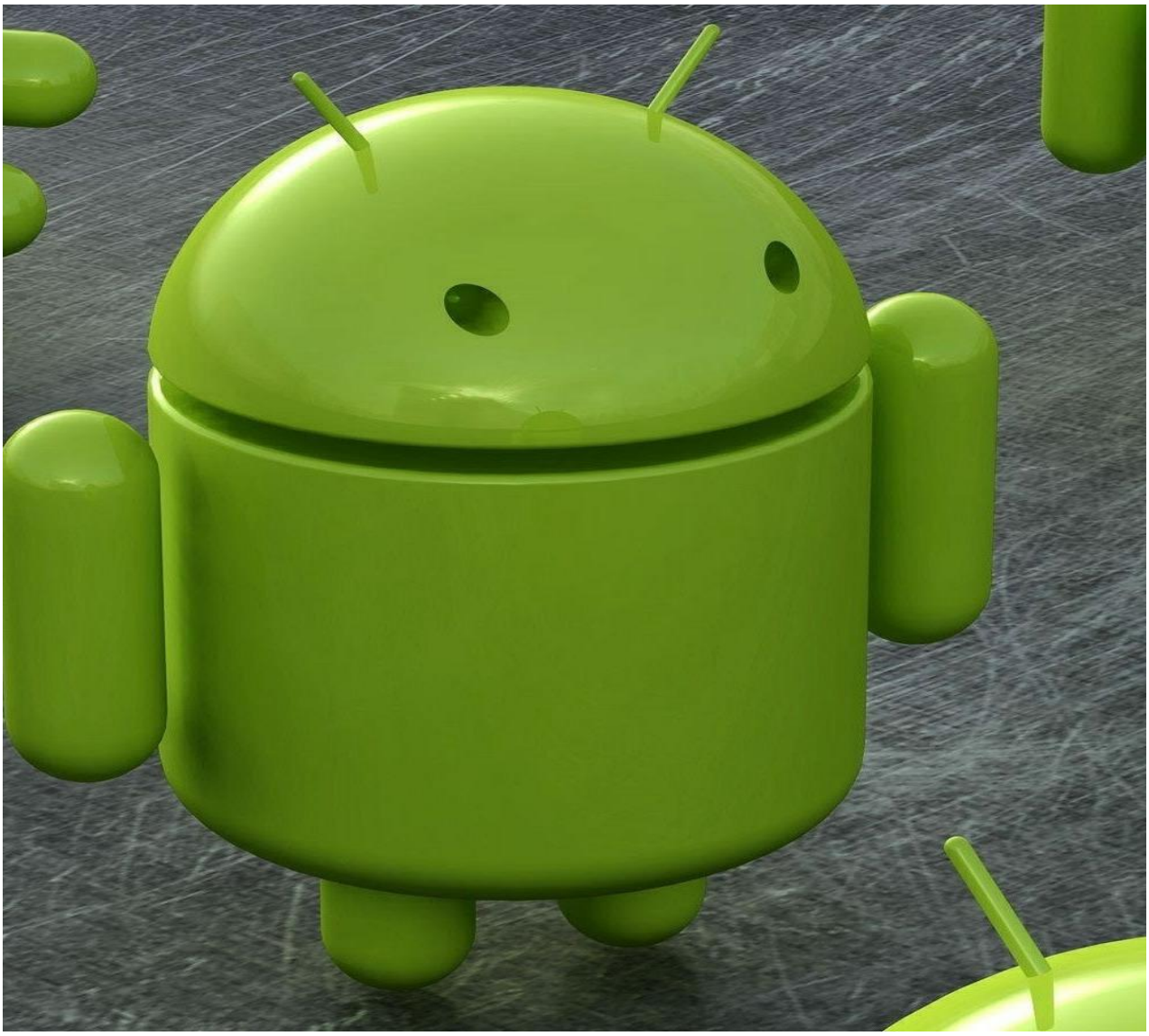


实验九

网络访问&Web 服务开发

【实验目的】

1. 熟练使用 HttpURLConnection 访问 WebService

2. 熟悉使用多线程以及 Handler 更新 UI

3. 熟悉使用 XmlPullParser 解析 xml 文档数据

4. 了解 RecyclerView 控件的使用

5. （可选）使用 Ksoap2 来访问 WebService

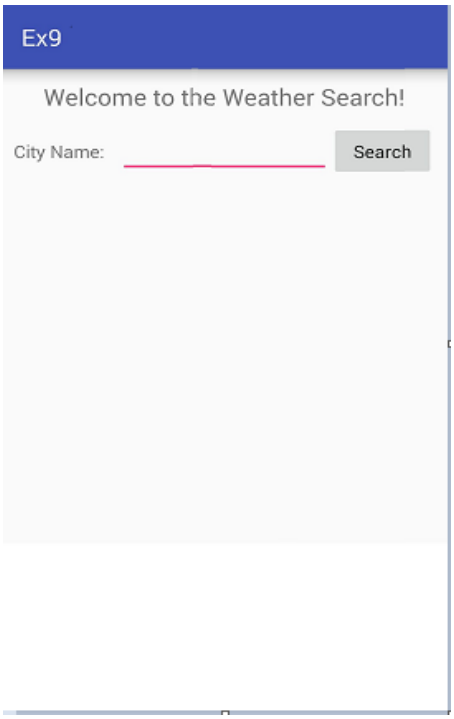
6. （可选）了解并掌握 SOAP 和 WSDL 相关基础

注意：可选内容与实验内容实现效果完全一致，只是实现方式不同，所以同学们可以二选一种方法来进行实验。

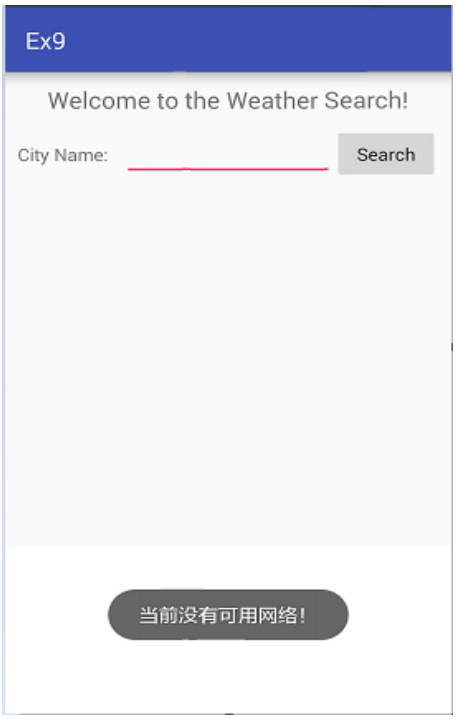
【实验内容】

实现一个简单的天气查询应用，具体要求如下：

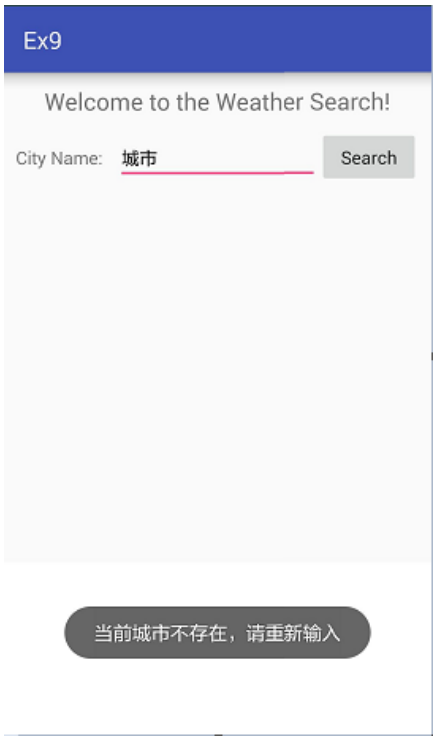
1）该界面为应用启动之后的主界面



2）点击 Search 按钮后（若网络不可用）

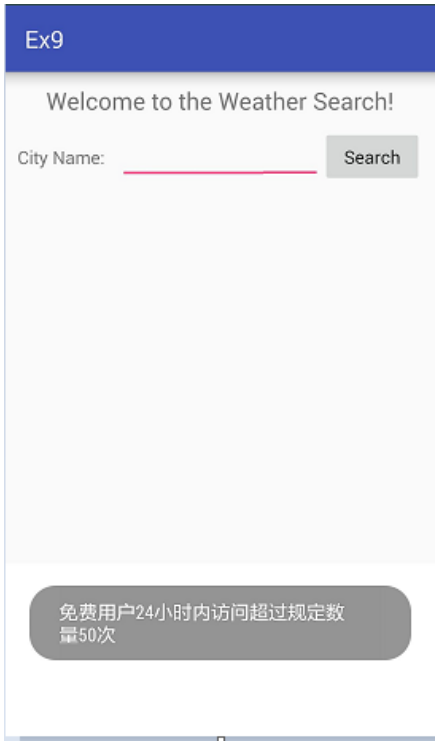


3）输入城市名 Search（网络可用且城市名正确）4）输入城市名 Search（城市名不正确）



5）快速点击按钮（二次查询<600ms）

6）查询达到上限 50 次



【参考内容】

1、实验 WebService 地址

（1）实验中所使用的 WebService 地址为：

http://ws.webxml.com.cn/WebServices/WeatherWS.asmx?op=getWeather

在浏览器打开实验需要的 WebService 网站可以看到截图如下：

可以看到，查询需要用到两个参数 theCityCode（默认参数为上海）和 theUserID，如果使用免费的 WebService，其中 theUserID 置空即可，输入后点击调用，查看返回值：

可以看到其中返回的数据为 xml 文件格式，我们需要用 XmlPullParser 提取我们需要的信息。

（2）有兴趣的同学可以在网站上看一下如何免费使用 WebService：

http://www.webxml.com.cn/zh_cn/web_services.aspx

PS：免费用户 24 小时内查询不超过 50 次并且获取二次数据大于间隔 600ms

（如果测试时超过上限更换一下网络 IP 重新测试即可）

2、网络访问

在本次实验中使用 HttpURLConnection 实现网络访问：

（1） 在 manifest 中添加网络访问权限：

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

（2） 判断是否有可用网络：使用 ConnectivityManager 获取手机所有连接管理对象，使用 manager 获取 NetworkInfo 对象，最后判断当前网络状态是否为连接状态即可。

（3） 定义我们需要用到的 WebService 地址：

```
private static final String url = "http://ws.webxml.com.cn/WebServices/WeatherWS.asmx/getWeather";
//网络服务地址
```

（4） 使用 HttpURLConnection 新建一个 http 连接，新建一个 URL 对象，打开连接即可，并且设置访问方法以及时间设置：

```
HttpURLConnection connection = null ;
try  {
    Log.i("key" ,"Begin the connection"  );
    connection = (HttpURLConnection) ((new URL(url.toString())
            .openConnection()));
    connection.setRequestMethod("POST");
    connection.setReadTimeout(8000);
    connection.setConnectTimeout(8000);
```

（5）将我们需要请求的字段以流的形式写入 connection 之中，这一步相当于将需要的参数提交到网络连接，并且请求网络数据（类似于 html 中的表单操作，将 post 数据提交到服务器）

```
 DataOutputStream out = new DataOutputStream(connection.getOutputStream());
// Log.i("key",city.getText().toString());
 String request = city .getText().toString();
 request = URLEncoder.encode(request, "utf-8" );//注意中文乱码解决

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

（6）网页获取 xml 转化为字符串：

```
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);
}
```

我们在 logcat 中查看一下 response，实际上就是将网站上的 xml 转化为 string 而已，便于下一步的 xml 数据解析

```
11-07 17:08:13.490 13799-14815/com.example.administrator.ex9 I/key: <?xml version="1.0" encoding="utf-8"?><ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://WebXml.com.cn/"> <string>广东 广州</string> <string>广州</string> <string>2350</string> <string>2016/11/07 16:50:52</string> <string>今日天气实况：气温：29℃；风向/风力：东南风 1级；湿度：51%</string> <string>紫外线强度：弱。空气质量：中。</string> <string>紫外线指数：弱，辐射较弱，涂擦SPF12-15、PA+护肤品。感冒指数：少发，无明显降温，感冒机率较低。穿衣指数：热，适合穿T恤、短薄外套等夏季服装。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，户外运动请注意防晒。空气污染指数：中，易感人群应适当减少室外活动。</string> <string>11月7日 多云</string> <string>19℃/28℃</string> <string>无持续风向微风</string> <string>1.gif</string> <string>1.gif</string> <string>11月8日 多云转阴</string> <string>15℃/24℃</string> <string>北风 4-5级转3-4级</string> <string>1.gif</string> <string>2.gif</string> <string>11月9日 小雨</string> <string>14℃/20℃</string> <string>无持续风向微风</string> <string>7.gif</string> <string>7.gif</string> <string>11月10日 小雨转阴</string> <string>13℃/17℃</string> <string>无持续风向微风</string> <string>7.gif</string> <string>2.gif</string> <string>11月11日 多云</string>
```

（7）关闭 connection：

```
finally   {
  if  (connection != null ) {
      connection.disconnect();
  }
```

（8）注：Android4.0 之后，http 请求需要开启子线程，然后由子线程执行请求，所以我们之前所写代码都是在子线程中完成的，并且使用 XmlPullParser 进行解析从而得到我们想要的数据：

```
private void    sendRequestWithHttpURLConnection() {
    new Thread(new Runnable() {
        @Override
        public void    run() {
            /*http请求操作*/
            /*XmlPullParser操作*/
            /*Message消息传递*/
        }
    }).start();

}
```

3、Handler 更新 UI

在之前的实验中我们已经知道，子线程中不能直接修改 UI 界面，需要中间人 handler 进行 UI 界面的修改：

```
private   Handler handler = new Handler(){
    public void   handleMessage(Message message){
        switch  (message.what){
            case UPDATE_CONTENT:
                //由子线程得到的字符串，对其进行处理
                //UI界面更新
                break;
            default :
                break;
        }
    }
};
```

Message 消息机制：负责在不同的线程之间进行交互处理，我们先定义消息类型：

```
private static final int       UPDATE_CONTENT = 0;
```

然后将我们需要的内容通过消息传递回来：

```
Message message = new Message();
message.what = UPDATE_CONTENT;
message.obj = parseXMLWithPull(response.toString());
handler.sendMessage(message);
```

4、XmlPullParser 解析 xml 文档

首先获取 XmlPullParser 对象实例，然后设置需要解析的字符串，最后按照 tag 逐个获取所需要的 string：

```
XmlPullParserFactory factory=XmlPullParserFactory.newInstance ();// 获取实例
XmlPullParser parser = factory.newPullParser();
parser.setInput(new StringReader(xml));// 设置所需要解析的string
```

```
int eventType = parser.getEventType();
while (eventType!=XmlPullParser.END_DOCUMENT){
    switch (eventType){
        case XmlPullParser.START_TAG:
            if ("string" .equals(parser.getName())){
                String str = parser.nextText();
              // Log.i("key",str);
                list.add(str);
            }
            break;
        case XmlPullParser.END_TAG:
            break;
        default :
            break;
    }
    eventType=parser.next();
}
```

注：由于我们获取的 xml 字符串是 string 类型的数组（ArrayofString），所以我们也可以将按照 string（Tag）获取的字符串储存到 ArrayList<String>中，然后在 Handler 中再进行处理，不断的将所需要的字符串进行分割处理，以用来更新 UI 界面。

注：异常处理部分：

1）城市名输入错误时：注意判断取回的字符串是否是“查询结果为空。…”

```
11-11 17:04:10.965 17755-17755/com.example.administrator.ex9 I/key: 查询结果为空。http://www.webxml.com.cn/
```

2）二次查询间隔<600ms 时：注意判断取回的字符串是否是“发现错误：免费用户不能使用高速访问。…”

11-11 21:13:01.760 17136-18216/com.example.administrator.ex9 I/key: <?xml version="1.0" encoding="utf-8"?><ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://WebXml.com.cn/"> <string>发现错误：免费用户不能使用高速访问。http://www.webxml.com.cn/</string></ArrayOfString>

3）达到上限 50 次：注意判断取回的字符串是否是“发现错误：免费用户 24 小时内访问超过规定数量。…”

11-11 21:20:14.510 20323-25092/com.example.administrator.ex9 I/key: <?xml version="1.0" encoding="utf-8"?><ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://WebXml.com.cn/"> <string>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/</string></ArrayOfString>

【可选实现方法】

1.阅读 WSDL 文件，获取 WebService 访问所需要的信息。关于 WSDL 语言的详细内容，可以阅读以下资料(建议同学们课后去阅读一下，加深对 WSDL 的理解)：

w3school 的 WSDL 教程：http://www.w3school.com.cn/wsdl/

Uderstanding WSDL: http://msdn.microsoft.com/en-us/library/ms996486.aspx

本实验所使用的 WebService 的 WSDL 文件的链接：

http://ws.webxml.com.cn/WebServices/WeatherWS.asmx?WSDL

通过阅读 WSDL 文档，将获取到以下的信息（其获取过程参阅扩展阅读）：

Webservice 的命名空间，url，操作名，soapAction，请求参数名，应答属性名

```
▼<wsdl:operation name="getWeather">
  ▼<wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
     <br /><h3>获得天气预报数据</h3><p>输入参数：城市/地区ID或名称，返回数据：一维字符串数组。</p><br />
   </wsdl:documentation>
   <wsdl:input message="tns:getWeatherSoapIn"/>
   <wsdl:output message="tns:getWeatherSoapOut"/>
 </wsdl:operation>

▼<wsdl:operation name="getWeather">
    <soap:operation soapAction="http://WebXml.com.cn/getWeather" style="document"/>
```

```
private static final String NAMESPACE = "http://WebXml.com.cn/";
private static final String URL = "http://ws.webxml.com.cn/WebServices/WeatherWS.asmx";
private static final String OPERATION_NAME = "getWeather";
private static final String SOAP_ACTION = "http://WebXml.com.cn/getWeather";
```

2.添加 Ksoap2 第三方类库到工程类库中：

导入 jar 包，导入方法：
http://jingyan.baidu.com/article/e6c8503c7190b7e54f1a1893.html

3.使用 Ksoap2 访问 WebService，并获取结果（注意，WebService 访问需要另外新启子线程执行，方法与前面描述一致）

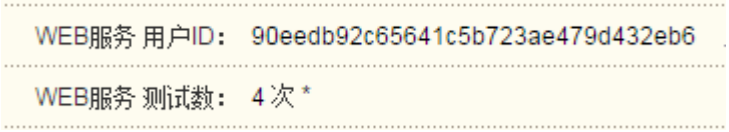
（1）.创建请求对象，并设置参数：

```
//创建请求对象
SoapObject request= new SoapObject(NAMESPACE,OPERATION_NAME);
//添加请求参数
request.addProperty("theCityCode",City);
request.addProperty("theUserID","90eedb92c65641c5b723ae479d432eb6");
```

注意：使用 Soap 访问天气服务在该网站为收费功能，如果要试用，需要注册一个账号来获取用户 ID。

注册网址：http://www.webxml.com.cn/user/user_reg.aspx

注册后可以在 http://www.webxml.com.cn/user/my_user_space.aspx 查看用户 ID 和剩余测试次数（使用 soap 在一个时间段内查询同一个城市不会减少测试次数）



（2）.创建 SoapSerializationEnvelope 对象，并设置请求对象：

```
//生成调用WebService方法的SOAP请求信息
SoapSerializationEnvelope envelope = new SoapSerializationEnvelope(SoapEnvelope.VER11);
//兼容.Net-Serivces的默认编码
envelope.dotNet = true;
//设置发送对象
envelope.setOutputSoapObject(request);
```

（3）.创建 HttpTransportSE 对象，发送请求：

```
//创建HttpTransportSE对象
HttpTransportSE transport = new HttpTransportSE(URL);
try {
    //设置soapAction header并发送请求，获取结果
    transport.call(SOAP_ACTION, envelope);
} catch (Exception e) {
    e.printStackTrace();
}
```

（4）.获取结果

```
//获取返回envelope的body信息
SoapObject bodyIn=(SoapObject)envelope.bodyIn;
//获取getWeatherResult属性
SoapObject detail = (SoapObject)bodyIn.getProperty("getWeatherResult");
```

假若我们输出 detail.toString()，会得到如下结果

```
anyType{string=直辖市 北京; string=北京; string=792; string=2016/11/22 20:58:32; string=今日天气实况：气温：-2℃；风向/风力：
```

每个 string 字段后都是其中的一个元素，按数组顺序排列。例如我们想返回第 5 个元素 string=今日天气实况：... 则需使用

```
String result = detail.getPropertyAsString(4);
```

来获取。天气信息获取成功后，后续步骤不再赘述。

【拓展知识】

一、RecyclerView 控件使用

RecyclerView 是 support-v7 包中的新组件，是一个强大的滑动组件，与经典的 ListView 相比，同样拥有 item 回收复用的功能。RecyclerView 是 ListView 的升级版，最为突出的就是其拓展性极强，并且灵活性高。

（1）导入 RecyclerView 的 jar 包，导入方法：

http://jingyan.baidu.com/article/e6c8503c7190b7e54f1a1893.html

（2） RecylerView 的设置：

```
recyclerView   = (RecyclerView) findViewById(R.id.weather_horizontal);
LinearLayoutManager layoutManager = new LinearLayoutManager(this );
layoutManager.setOrientation(LinearLayoutManager.HORIZONTAL);// 设置为水平
recyclerView .setLayoutManager(layoutManager);
```

（3）Adapter 的构造：

```
package com.example.administrator.ex9;

import java.util.ArrayList;
import java.util.List;
import android.R.integer;
import android.content.Context;
import android.graphics.Bitmap;
import android.support.v7.widget.RecyclerView;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

public class WeatherAdapter extends RecyclerView.Adapter<WeatherAdapter.ViewHolder>{

    private ArrayList<Weather> weather_list;
    private LayoutInflater mInflater;

    public interface OnItemClickLitener
    {
        void onItemClick(View view, int position,Weather item);
    }

    private OnItemClickLitener mOnItemClickLitener;

    public void setOnItemClickLitener(OnItemClickLitener mOnItemClickLitener)
    {
        this.mOnItemClickLitener = mOnItemClickLitener;
    }
```

```
    public WeatherAdapter(Context context,ArrayList<Weather> items) {
        super();
        weather_list = items;
        mInflater = LayoutInflater.from(context);
    }

    @Override
    public ViewHolder onCreateViewHolder(ViewGroup viewGroup, int i) {
        View view = mInflater.inflate(R.layout.weather_item, viewGroup, false);
        ViewHolder holder = new ViewHolder(view);
        holder.Date = (TextView)view.findViewById(R.id.date);
        holder.Weather_description =(TextView)view.findViewById(R.id.weather_description);
        holder.Temperature = (TextView)view.findViewById(R.id.temperature);
        return holder;
    }

    @Override
    public void onBindViewHolder(final ViewHolder viewHolder,    final int i) {
        viewHolder.Date.setText(weather_list.get(i).getDate());

viewHolder.Weather_description.setText(weather_list.get(i).getWeather_description());
        viewHolder.Temperature.setText(weather_list.get(i).getTemperature());
        if (mOnItemClickLitener != null)
        {
            viewHolder.itemView.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    // TODO Auto-generated method stub
                mOnItemClickLitener.onItemClick(viewHolder.itemView, i,weather_list.get(i));
                }
            });
        }
    }

    @Override
    public int getItemCount() {
        return weather_list.size();
    }

    public static class ViewHolder extends RecyclerView.ViewHolder{
        public ViewHolder(View itemView) {
```

```
                super(itemView);
            }
            TextView Date;
            TextView Weather_description;
            TextView Temperature;
        }
}
```

（3） Handler 中得到 weather_list 之后，绑定 adpter：

```
adapter = new WeatherAdapter(MainActivity.this ,weather_list );
recyclerView .setAdapter(adapter);
```

PS：RecyclerView 控件有更多灵活的用法，此处不一一介绍，希望大家做项目的时候可以进一步挖掘。

【检查内容】

1、布局显示是否正常（listview 等 其余控件摆放合理即可）

2、点击按钮时网络是否可用的判断

3、是否可以正常访问 WebService

4、温度 湿度 更新时间等必要信息是否显示正常

5、RecyclerView 控件界面展示（近五天天气）为拓展内容，不做要求。

6、城市名称是否正确，查询时间间隔等抛出异常，查询上限 50 次异常不做要求。

7、无论使用哪种方式实现，请在实验报告中说明。

【提交说明】

1、deadline：下一次实验课前一天晚上 12 点

2、提交 ftp：ftp://222.200.185.18:1890/，作业提交文件夹中对应的文件夹下

3、命名与目录结构要求：

附件命名及格式要求：学号_姓名_labX.zip（姓名中文拼音均可）

重复提交命名格式要求：学号_姓名_labX_Vn.zip

目录结构：

```
14331111_huashen_lab1 --
                      |
                      --  lab1实验报告.pdf
                      |
                      --  lab1_code（包含项目代码文件）
```

其中项目代码文件为项目文件夹，提交之前先 clean。