

手机平台应用开发实验报告

学号: 14331098

班级: 周五下午班

姓名: 黄建武

实验名称: Lab9

一、参考资料

1. [Android官方开发者文档](#)
2. [和风天气API](#)
3. [URLConnection进行网络访问](#)
4. [ImageView获取网络图片](#)

二、实验环境

- 系统: Windows 10 64位
- 工具: Android Studio 2.1.3
- JDK版本: 1.8
- minSdk: 19
- targetSdk: 24

三、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 org.json 解析 json 数据
4. 了解 RecyclerView 控件的使用

四、实验步骤

1. 阅读实验要求文档以及对应的课件PPT。
2. 运行Demo程序，了解实验要求细节。
3. 首先实现界面上方的布局，输入框的layout_weight设为1，查找合适的天气API，使用URLConnection进行网络访问，创建新的线程进行网络访问，使用BufferedReader读取连接的输入流，进行输出调试，确认正确地获取到服务器返回的数据。
4. 实现生活指数的布局，使用ListView，自定义适配器，并且为ListView添加适配器，处理返回的json数据，获取生活指数部分并使用Handler更新ListView。
5. 实现今日天气信息部分的布局，以及处理对应部分的json数据并使用Handler更新UI，数据中有一个属性是天气代号，获取该代号后向服务器请求对应的天气图片，并将输入流解码为Bitmap，在页面上显示出来。
6. 实现显示未来几天天气的部分，使用RecyclerView实现布局，自定义适配器，实验文档的PDF里是自定义一个Weather类，然后适配器使用的数据结构是ArrayList，而我使用的api中未来一周的天气是一个Json数组，因此数据结构使用JsonArray，仿照实验文档进行相应的更改，然后在Handler里为RecyclerView添加适配器。
7. 对已完成的部分进行调试。
8. 完善逻辑，正确处理城市名为空，城市不存在，点击过快，查询外国城市等情况。两次点击间隔过小Toast提醒，因为所用的天气api没有速度限制，因此记录上一次请求的时间，判断前后两次请求的间隔；所用api可以查询外国天气，但是返回数据为全英文而且没有生活指数部分，故设置为只能查询国内天气。
9. 完善页面样式。

主要代码部分如下：

```
// 点击search按钮
search.setOnClickListener(new View.OnClickListener() {
```

```

@Override
public void onClick(View v) {
    citystr = cityname.getText().toString().trim();
    if (citystr.isEmpty()) { // 城市名为空
        showToast(R.string.emptycityname);
        return;
    }
    if (!hasNetwork()) { // 没有联网
        showToast(R.string.nonetwork);
        return;
    }
    long current = System.currentTimeMillis(), tmp = lastclick;
    lastclick = current;
    if (current - tmp < 600) // 请求过快
        showToast(R.string.toofast);
    else
        getWeather();
}
});

```

```

// 获取天气数据
private void getWeather() {
    new Thread(new Runnable() {
        @Override
        public void run() {
            try {
                URL url = new URL(HTTPURL + URLEncoder.encode(citystr, "utf-8"));
                HttpURLConnection connection = (HttpURLConnection) url.openConnection();
                connection.setRequestProperty("apikey", "48b13529ff54895d6837e64403a84b49");
                connection.setDoInput(true);
                if (connection.getResponseCode() == 200) {
                    InputStream is = connection.getInputStream();
                    BufferedReader reader = new BufferedReader(new InputStreamReader(is));
                    String line, json = "";
                    while ((line = reader.readLine()) != null)
                        json += line;
                    is.close();
                    connection.disconnect();
                    decodeJson(json);
                }
                else
                    showToast(R.string.timeout);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }).start();
};

```

```

// 处理json数据
private void decodeJson(String json) throws JSONException {
    JSONObject obj = new JSONObject(json);
    obj = obj.getJSONArray("HeWeather data service 3.0").getJSONObject(0);
    Message msg1 = new Message(), msg2 = new Message(), msg3 = new Message();
    if (obj.getString("status").equals("ok")) {
        msg1.what = UPDATESUGGESTION;
        if (!obj.has("suggestion")) { // 国外城市
            msg1.what = NOTCHINA;
            handler.sendMessage(msg1);
            return;
        }
        // 更新生活指数
        msg1.what = UPDATESUGGESTION;
        msg1.obj = obj.getJSONObject("suggestion");
    }
}

```

```

        handler.sendMessage(msg1);

        // 更新今日天气信息
        JSONArray dailyforecast = obj.getJSONArray("daily_forecast");
        msg2.what = UPDATENOW;
        // 今日数据里没有更新时间以及当前温度，手动添加
        msg2.obj = dailyforecast.getJSONObject(0)
        .put("update", obj.getJSONObject("basic").getJSONObject("update").getString("loc").substring(5))
        .put("nowtmp", obj.getJSONObject("now").getString("tmp"));
        handler.sendMessage(msg2);

        // 更新未来6天天气信息
        dailyforecast.remove(0);
        msg3.what = UPDATEDAILY;
        msg3.obj = dailyforecast;
        handler.sendMessage(msg3);
    } else if (obj.getString("status").equals("unknown city")) { //城市名不存在
        msg3.what = CITYNOTEXISTS;
        handler.sendMessage(msg3);
    }
}
}

```

```

// 获取天气图片
private void getIcon(final String code_d) {
    new Thread(new Runnable() {
        @Override
        public void run() {
            try {
                bitmap = null;
                URL url = new URL(ICONURL + code_d + ".png");
                HttpURLConnection connection = (HttpURLConnection) url.openConnection();
                connection.setDoInput(true);
                if (connection.getResponseCode() == 200) {
                    InputStream is = connection.getInputStream();
                    bitmap = BitmapFactory.decodeStream(is);
                    is.close();
                    connection.disconnect();
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
            Message msg = new Message();
            msg.what = UPDATEICON;
            handler.sendMessage(msg);
        }
    }).start();
};

```

```

// 绑定RecyclerView
@Override
public void onBindViewHolder(final ViewHolder viewHolder, final int i){
    try {
        JSONObject obj = dailyforecast.getJSONObject(i);
        viewHolder.Date.setText(obj.getString("date").substring(5));
        viewHolder.Weather.setText(obj.getJSONObject("cond").getString("txt_d"));
        JSONObject tmp = obj.getJSONObject("tmp");
        String t = tmp.getString("min") + "°C/" + tmp.getString("max") + "°C";
        viewHolder.Temp.setText(t);
    } catch (JSONException e) {
        e.printStackTrace();
    }
}
}

```

五、实验结果



六、实验问题

1. 一开始在构建天气查询的url的时候没有把输入的城市名以utf-8编码，导致请求失败，未能获取到数据，使用 `URLEncoder.encode(citystr, "utf-8")`能正确发出请求。
2. 发出请求后使用 `getInputStream`获取输入流，然后使用 `getContentLength`获取数据长度，然后一次性读取全部字节到一个byte数组中，再使用该数组构造String后对该json字符串进行处理，可是有时在处理json的时候程序会崩，显示字符串不符合json格式，但有时又能正常解析，上网查阅后发现 `getContentLength`与网络状况有关，网络不佳时获取到的数据长度会小于真实长度，于是将长度输出，果然，有时四千多字节有时才两千多字节，两千多字节时字符串不完整因此解析错误，最后改用以下方法读取：

```
InputStream is = connection.getInputStream();
BufferedReader reader = new BufferedReader(new InputStreamReader(is));
String line, json = "";
while ((line = reader.readLine()) != null)
    json += line;
is.close();
```

3. 更新UI时老是忘了需要在Handler中更新，总是获取数据后就直接更新，没报错也没效果。
4. 因为页面上的内容较多，经过考虑后决定将未来几天的天气信息显示在生活指数下，向上滑动才能看到未来的天气信息，为了考虑到向上滑动后如果要再次查询需要再滚上去，用户体验不是很好，因此将输入框下方所有控件放在一个ScrollView中，这样滑动的时候输入框和按钮位置不变。当我加上未来几天的天气信息后，更新UI，页面没用

什么变化，以为是适配器或者更新的时候出错了，debug了好久，最后发现RecyclerView的高度原先是设置为wrap_content，而当我设置固定高度时，ScrollView可以滑动，并且能够显示出RecyclerView。

七、实验总结

本次实验难度适中，我使用了URLConnection进行网络访问，然后对获取到的json数据进行解析和处理，有过python爬虫的经验，使用java进行网络访问也比较顺利，只是不太习惯Android中的网络访问一定要新的线程中执行，经过本次实验，掌握了URLConnection进行网络访问的方法，使用getInputStream的时候就会调用connect函数，在读取字符串数据的时候不要一口气读取整个输入流，网络状况不佳的时候读取的数据可能不完整。这次实验中对Handler还是不太熟练，老是忘了要使用Handler更新UI，以后的实验需要注意。