使用 Volley 访问 JSON 资源

知识解析

如果访问的资源是 JSON 格式的数据，则可以使用 JsonObjectRequest 或者 JsonArrayRequest 来访问网络。它们用法基本一致，不过是返回的数据类型有区别而已。

它的构造器：

JsonObjectRequest(int method, String url, JSONObject jsonRequest, Response.Listener<JSONObject> listener, Response.ErrorListener errorListener)，其中，第 3 个参数是用 JSONObject 封装的需要提交的参数，可以为 null

接收到数据后，可以通过 org.json 包中的相关类来对 json 数据进行处理

可以尝试去访问下面网址中的天气接口，它返回 JSON 格式的天气预报数据。

http://wthrcdn.etouch.cn/weather_mini?citykey=101010100

传递表单参数

如果要给网络资源传递参数，可以通过覆盖 XXXRequest 的父类 Request 中的 Map<String, String> getParams() 方法，将需要传递的参数放到 Map 中返回即可。

功能演示

```
温度：14感冒指数：天气较凉，较易发生感冒，请适当增加衣服。体质较弱的朋友尤其应该注意防护。
{"fengxiang":"无持续风向","fengli":"微风级","high":"高温 20℃","type":"阵雨","lo ⏎
w":"低温 14℃","date":"30日星期三"}
{"fengxiang":"北风","fengli":"4-5级","high":"高温 22℃","type":"晴","low": ⏎
"低温 11℃","date":"1日星期四"}
{"fengxiang":"无持续风向","fengli":"微风级","high":"高温 25℃","type":"晴","low ⏎
":"低温 12℃","date":"2日星期五"}
{"fengxiang":"无持续风向","fengli":"微风级","high":"高温 24℃","type":"晴","low ⏎
":"低温 12℃","date":"3日星期六"}
{"fengxiang":"无持续风向","fengli":"微风级","high":"高温 22℃","type":"晴","low ⏎
":"低温 13℃","date":"4日星期天"}
```

操作实践

第一步，定义 JsonObjectRequest 对象：

JsonObjectRequest request;

第二步通过 Volloy 实例化 RequestQueue 对象

RequestQueue queue = Volley.newRequestQueue(this);

第三步开始请求

```
request = new JsonObjectRequest(Method.GET, url, null, new
Response.Listener<JSONObject>() {

                @Override
                public void onResponse(JSONObject response) {
                    // TODO Auto-generated method stub
                    try {
                        JSONObject obj = response.getJSONObject("data");
                        int wendu = obj.getInt("wendu");
                        String ganmao = obj.getString("ganmao");
                        JSONArray array = obj.getJSONArray("forecast");
                        Log.i("当前天气：", "温度：" + wendu + "感冒指数：" + ganmao);
                        for (int i = 0; i < array.length(); i++) {
                            Log.i("天气", array.getJSONObject(i).toString());
                        }
                    } catch (JSONException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                    }
                }
            }, new Response.ErrorListener() {

                @Override
                public void onErrorResponse(VolleyError error) {
                    // TODO Auto-generated method stub

                }
            });

            queue.add(request);
```

职业素质

平时在开发Android应用的时候不可避免地都需要用到网络技术，而多数情况下应用程序都会使用HTTP协议来发送和接收网络数据。Android系统中主要提供了两种方式来进行HTTP通信，HttpURLConnection和HttpClient，几乎在任何项目的代码中我们都能看到这两个类的身影，使用率非常高。

不过HttpURLConnection和HttpClient的用法还是稍微有些复杂的，如果不进行适当封装的话，很容易就会写出不少重复代码。于是乎，一些Android网络通信框架也就应运而生，比如说AsyncHttpClient，它把HTTP所有的通信细节全部封装在了内部，我们只需要简单调用几行代码就可以完成通信操作了。