Android程序设计

数据格式

2018.6.20

isszym　sysu.edu.cn

http://www.runoob.com/w3cnote/android-tutorial-socket1.html

目录

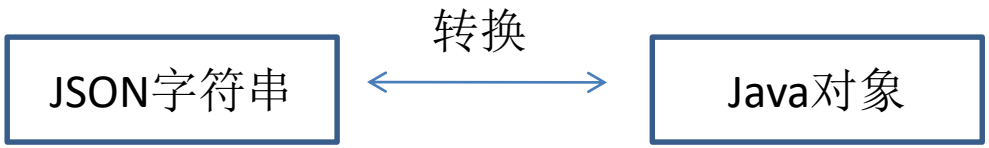
- Json格式
- XML格式
- SOAP格式
- ProtoBuf格式

Json格式

- Json数据格式

```
{  "firstName": "John",
   "lastName": "Smith",
   "sex": "male",
   "age": 25,
   "address":
   {  "streetAddress": "21 2nd Street",
      "city": "New York",
      "state": "NY",
      "postalCode": "10021"
   },
   "phoneNumber":
   [{  "type": "home",
        "number": "212 555-1234"
      },
      {"type": "fax",
       "number": "646 555-4567"
      }
   ]
}
```

JSON数据格式是JavaScript使用的是中数据格式，可以用于在不同系统中传递数据。它比XML格式效率更高，比ProtoBuf格式效率低，但是它很容易转换为JavaScript中的对象和数据，因此，使用最普遍。



- 第一种转换方法 Json1 不需要加入包

```
class Phone {
    String type, number;
    void setType(String val) { type = val; }
    String getType() { return type;}
    void setNumber(String val) { number = val; }
    String getNumber() { return number;}
}

class Address {
    String streetAddress,city,state,postalCode;
    void setStreetAddress(String val) { streetAddress = val; }
    String getStreetAddress() { return streetAddress; }
    void setCity(String val) {city = val; }
    String getCity() {return city;}
    void setState(String val) { state = val; }
    String getState() { return state; }
    void setPostalCode(String val) {  postalCode = val; }
    String getPostalCode() { return  postalCode; }
}
```

```
class User {
    String firstName;       String lastName;
    String sex;             int age;
    Address address;        List<Phone> phones;
    User(){}
    User(String firstName, String lastName, String sex, int age) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.sex = sex;
        this.age = age;
    }
    public void setFirstName(String val) { firstName = val; }
    public String getFirstName() { return firstName; }
    public void setLastName(String val) { lastName = val; }
    public String  getLastName() { return lastName; }
    public void setSex(String val) { sex = val; }
    public String getSex() { return sex; }
    public void setAge(int val) { age = val; }
    public int getAge(){ return age; }
    public void setAddress(Address val) { address = val;}
    public Address getAddress() { return address;}
    public void setPhoneNumber(List<Phone> phones) { this.phones = phones; }
    public List<Phone> getPhoneNumber() { return phones;}
    public String toString() {
        return "name: " + firstName + " " + lastName+", age: " + age;
    }
}
```

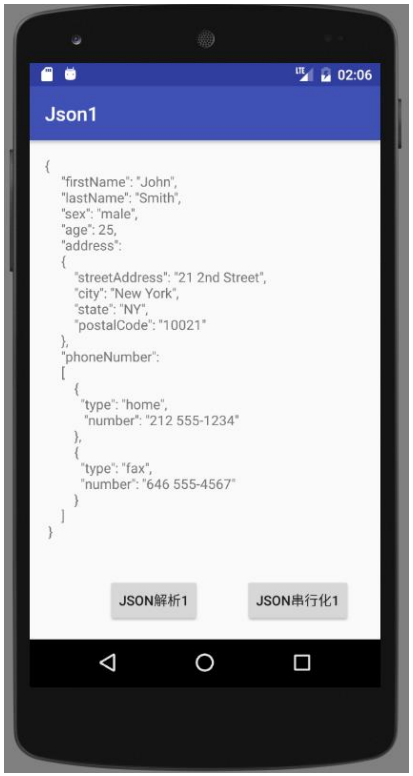
```
import org.json.JSONArray;
import org.json.JSONObject;                          /*安卓相关包，不用另外加入包*/
public class MainActivity extends AppCompatActivity {
    String user1 = "{\n" +
            "     \"firstName\": \"John\",\n" +
            "     \"lastName\": \"Smith\",\n" +
            "     \"sex\": \"male\",\n" +
            "     \"age\": 25,\n" +
            "     \"address\": \n" +
            "     {\n" +
            "         \"streetAddress\": \"21 2nd Street\",\n" +
            "         \"city\": \"New York\",\n" +
            "         \"state\": \"NY\",\n" +
            "         \"postalCode\": \"10021\"\n" +
            "     },\n" +
            "     \"phoneNumber\": \n" +
            "     [\n" +
            "         {\n" +
            "           \"type\": \"home\",\n" +
            "            \"number\": \"212 555-1234\"\n" +
            "         },\n" +
            "         {\n" +
            "           \"type\": \"fax\",\n" +
            "           \"number\": \"646 555-4567\"\n" +
            "         }\n" +
            "     ]\n" +
            " }";
    User user;
```

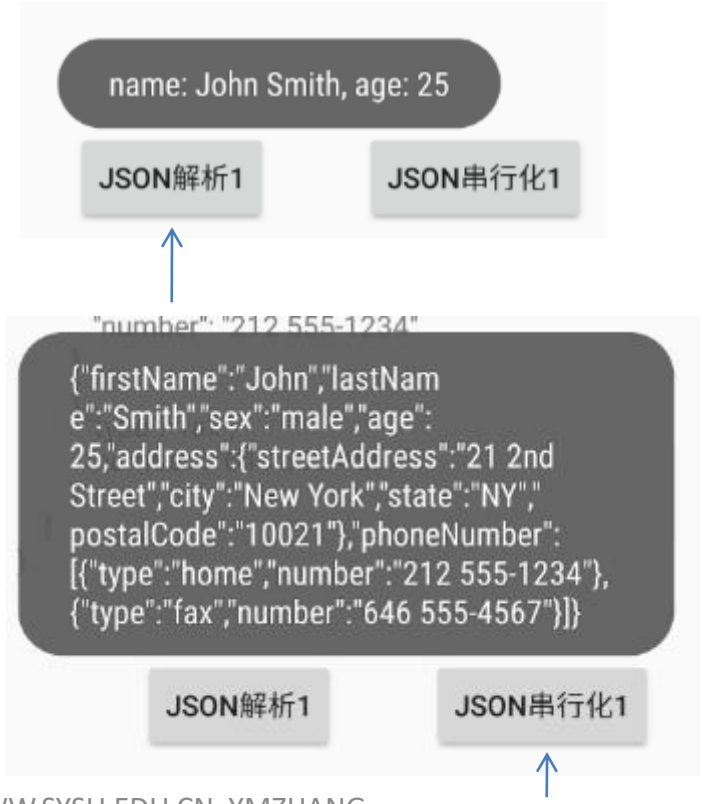
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    TextView textView=(TextView)findViewById(R.id.textView);
    textView.setText(user1);
    System.out.print(user1);
}

public static User parserJSON(String str) {
    JSONObject userObj=null;
    try {
        userObj = new JSONObject(str);
    } catch (Exception ex) {
        ex.printStackTrace();
    }
    User user = new User();
    user.setFirstName(userObj.optString("firstName"));
    user.setLastName(userObj.optString("lastName"));
    user.setSex(userObj.optString("sex"));
    user.setAge(userObj.optInt("age"));

    JSONObject addressObj = userObj.optJSONObject("address");
```

```
        Address address = new Address();
        address.setStreetAddress(addressObj.optString("streetAddress"));
        address.setCity(addressObj.optString("city"));
        address.setState(addressObj.optString("state"));
        address.setPostalCode(addressObj.optString("postalCode"));

        user.setAddress(address);

        JSONArray phoneNumber = userObj.optJSONArray("phoneNumber");

        List<Phone> phones = new ArrayList<>();
        for (int i = 0; i < phoneNumber.length(); i++) {
            Phone phone = new Phone();
            JSONObject phoneObj = phoneNumber.optJSONObject(i);
            phone.setType(phoneObj.optString("type"));
            phone.setNumber(phoneObj.optString("number"));
            phones.add(phone);
        }
        user.setPhoneNumber(phones);

        return user;
    }
```

```
public static String toJSON(User user){
    JSONObject userObj = new JSONObject();
    try { userObj.put("firstName", user.getFirstName());
        userObj.put("lastName", user.getLastName());
        userObj.put("sex", user.getSex());
        userObj.put("age", user.getAge());

        JSONObject addressObj = new JSONObject();
        Address address = user.getAddress();
        addressObj.put("streetAddress", address.getStreetAddress());
        addressObj.put("city", address.getCity());
        addressObj.put("state", address.getState());
        addressObj.put("postalCode", address.getPostalCode());

        userObj.put("address", addressObj);

        List<Phone> phoneNumber = user.getPhoneNumber();
        JSONArray phoneArrayObj = new JSONArray();
        for (int i = 0; i < phoneNumber.size(); i++) {
            JSONObject phoneObj = new JSONObject();
            phoneObj.put("type", phoneNumber.get(i).getType());
            phoneObj.put("number", phoneNumber.get(i).getNumber());
            phoneArrayObj.put(phoneObj);
        }
        userObj.put("phoneNumber", phoneArrayObj);
    } catch(Exception ex){
        ex.printStackTrace();
    }
    return userObj.toString();
}
```

```
    public void jsonSerialize1(View vw) {
        String user1=toJSON(user);
        Toast.makeText(this, user1, Toast.LENGTH_LONG).show();
    }

    public void jsonParse1(View vw) {
      user = parserJSON(user1);
      Toast.makeText(this, user.toString(),Toast.LENGTH_LONG).show();
    }
}
```





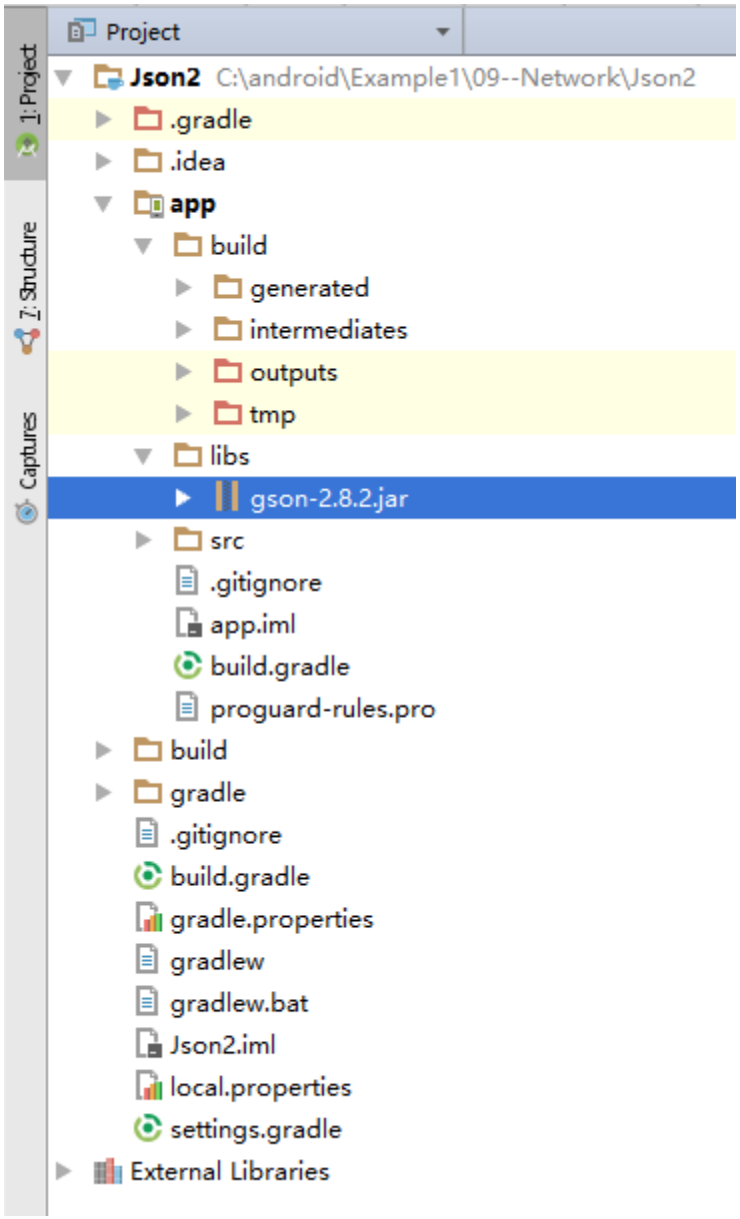
activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        android:id="@+id/textView" />
    <Button
        android:text="Json串行化1"
        android:onClick="jsonSerialize1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/button"/>
    <Button
        android:text="Json解析1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/button2"
        android:onClick="jsonParse1" />
</RelativeLayout>
```

- # 第二种转换方法　Json2

这种方法采用了Java的反射技术，使转换程序更简洁，但是需要引入外部包gson-2.8.2.jar

```
import com.google.gson.Gson;
import com.google.gson.JsonSyntaxException;
import com.google.gson.reflect.TypeToken;

 class Phone {   /* 与前一种方法相同 */
      …
 }

 class Address { /* 与前一种方法相同 */
     …
 }

 class User {    /* 与前一种方法相同 */
     …
 }
```



粘贴一个包进来，右键点出菜单“Add As a Library”

```
class GsonTools {
    /**TODO 转换为json字符串
     * @param src  要转换成json格式的 对象
     * @return
     */
    public static String createJsonString(Object src) {
        Gson gson = new Gson();
        String jsonString = gson.toJson(src);
        return jsonString;
    }
    /**TODO 转换为指定的 对象
     * @param jsonString
     * @param type 指定对象的类型 ， 即 T.class
     * @return
     */
    public static <T> T getObject(String jsonString, Class<T> type) {
        T t = null;
        try {
            Gson gson = new Gson();
            t = gson.fromJson(jsonString, type);
        } catch (JsonSyntaxException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return t;
    }
```

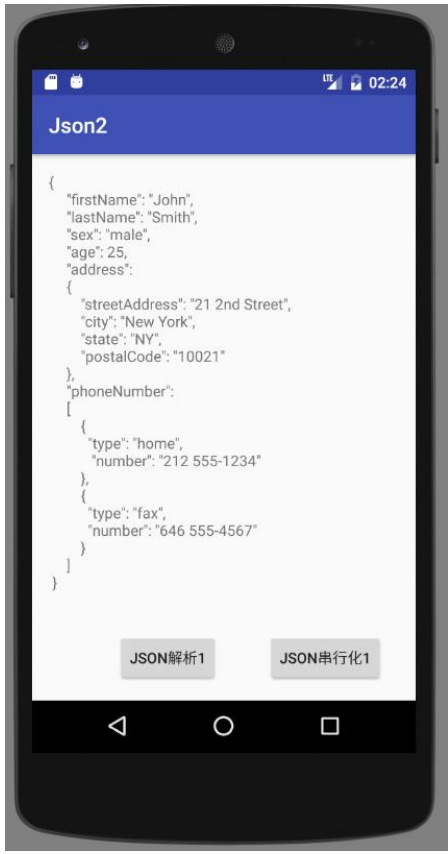
```java
/**得到 一个List<T>集合
 * @param jsonString
 * @param type  T的类型
 */
public static <T> List<T> getList(String jsonString, Class<T> type) {
    List<T> list = new ArrayList<T>();
    Gson gson = new Gson();
    list = gson.fromJson(jsonString, new TypeToken<List<T>>() {
    }.getType());
    return list;
}
/**TODO 得到一个List<T> 的集合
 * @param jsonString json字符串
 * @param type  数组的类型 , 即T[].class
 */
public static <T> List<T> StringTolist(String jsonString,Class<T[]>type){
    T[] list = null;
    try {
        Gson gson = new Gson();
        list = gson.fromJson(jsonString, type);
    } catch (JsonSyntaxException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    return Arrays.asList(list);
}
```

```
    /**把json字符串转换为 String 集合
     * @param jsonString
     */
    public static List<String> getStrings(String jsonString) {
        List<String> list = new ArrayList<String>();
        Gson gson = new Gson();
        new TypeToken<List<String>>(){}.getType();
        list = gson.fromJson(jsonString, new TypeToken<List<String>>() {
        }.getType());
        return list;
    }
    /**TODO 将json数据解析为Map<String,Object>集合
     * @param jsonString
     */
    public static List<Map<String, Object>> getMaps(String jsonString) {
        List<Map<String, Object>>list=new ArrayList<Map<String,Object>>();
        Gson gson = new Gson();
        list = gson.fromJson(jsonString,
                new TypeToken<List<Map<String, Object>>>() {
                }.getType());
        return list;
    }
}
```

```
public class MainActivity extends AppCompatActivity {
    String user1 =…;  /* 与前一种方法相同 */
    String phones =
            "     [\n" +
            "         {\n" +
            "           \"type\": \"home\",\n" +
            "            \"number\": \"212 555-1234\"\n" +
            "         },\n" +
            "         {\n" +
            "           \"type\": \"fax\",\n" +
            "           \"number\": \"646 555-4567\"\n" +
            "         }\n" +
            "     ]\n";
    User user;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        TextView textView=(TextView)findViewById(R.id.textView);
        textView.setText(user1);
    }
    public void jsonSerialize1(View vw) {
        String user1=GsonTools.createJsonString(user);
        Toast.makeText(this, user1,
                Toast.LENGTH_LONG).show();
    }
```

```
public void jsonParse1(View vw) {
     user = GsonTools.getObject(user1,User.class);
     List<Phone> ph= GsonTools.getList(phones,Phone.class);
     user.setPhoneNumber(ph);
     Toast.makeText(this, user.toString(),
             Toast.LENGTH_LONG).show();
 }
 public static User parserJSON(String str) {
     JSONObject userObj=null;
     try {
         userObj = new JSONObject(str);
     } catch (Exception ex) {
         ex.printStackTrace();
     }
     User user = new User();
     user.setFirstName(userObj.optString("firstName"));
     user.setLastName(userObj.optString("lastName"));
     user.setSex(userObj.optString("sex"));
     user.setAge(userObj.optInt("age"));

     JSONObject addressObj = userObj.optJSONObject("address");
```

```
        Address address = new Address();
        address.setStreetAddress(addressObj.optString("streetAddress"));
        address.setCity(addressObj.optString("city"));
        address.setState(addressObj.optString("state"));
        address.setPostalCode(addressObj.optString("postalCode"));

        user.setAddress(address);

        JSONArray phoneNumber = userObj.optJSONArray("phoneNumber");

        List<Phone> phones = new ArrayList<>();
        for (int i = 0; i < phoneNumber.length(); i++) {
            Phone phone = new Phone();
            JSONObject phoneObj
                        = phoneNumber.optJSONObject(i);
            phone.setType(phoneObj.optString("type"));
            phone.setNumber(phoneObj.optString("number"));
            phones.add(phone);
        }
        user.setPhoneNumber(phones);

        return user;
    }
}
```



activity_main.xml和执行结果与前一种方法相同。

XML格式

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Edited with XML Spy v2007 (http://www.altova.com) -->
<CATALOG>
        <PLANT>
                <COMMON>Bloodroot</COMMON>
                <BOTANICAL>Sanguinaria canadensis</BOTANICAL>
                <ZONE>4</ZONE>
                <LIGHT>Mostly Shady</LIGHT>
                <PRICE>$2.44</PRICE>
                <AVAILABILITY>031599</AVAILABILITY>
        </PLANT>
        <PLANT>
                <COMMON>Columbine</COMMON>
                <BOTANICAL>Aquilegia canadensis</BOTANICAL>
                <ZONE>3</ZONE>
                <LIGHT>Mostly Shady</LIGHT>
                <PRICE>$9.37</PRICE>
                <AVAILABILITY>030699</AVAILABILITY>
        </PLANT>
</CATALOG>
```

参考 解析

XML解析举例：

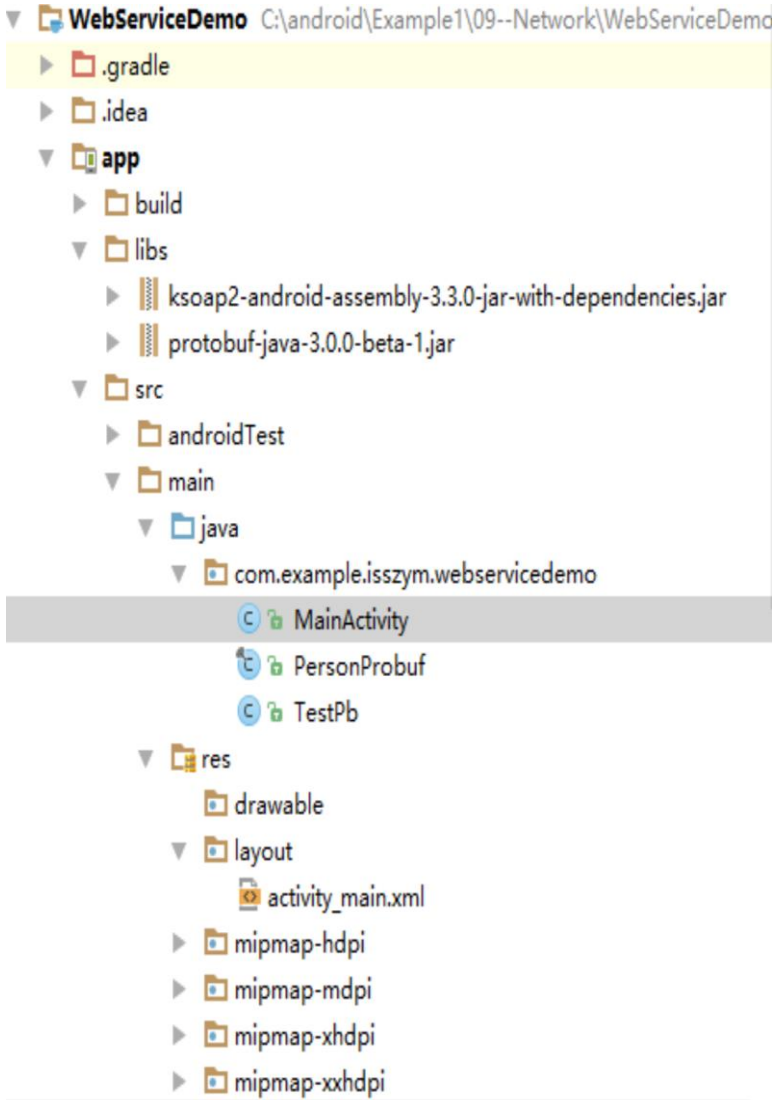
在Assets文件夹中模拟创建XML文件

```
<students>
    <student>
        <name sex="man">小明</name>
        <nickName>明明</nickName>
    </student>
    <student>
        <name sex="woman">小红</name>
        <nickName>红红</nickName>
    </student>
    <student>
        <name sex="man">小亮</name>
        <nickName>亮亮</nickName>
    </student>
</students>
```

```
public class Student {
    private String name;
    private String sex;
    private String nickName;
    public String getName() {
        return name;
    }
    public void setName(String name) {  this.name = name;  }
    public String getSex() {  return sex;  }
    public void setSex(String sex) {  this.sex = sex;   }
    public String getNickName() {  return nickName;  }
    public void setNickName(String nickName) {
            this.nickName = nickName;
    }
    @Override
    public String toString() {
        return "Student{" +
                "name='" + name + '\'' +
                ", sex='" + sex + '\'' +
                ", nickName='" + nickName + '\'' +
                '}';
    }
}
```

```
public List<Student> dom2xml(InputStream is) throws Exception {
    List<Student> list = new ArrayList<>();
    DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
    DocumentBuilder builder = factory.newDocumentBuilder();
    Document document = builder.parse(is);
    NodeList studentList = document.getElementsByTagName("student");
    for (int i = 0; i < studentList.getLength(); i++) {
        Node node_student = studentList.item(i);
        NodeList childNodes = node_student.getChildNodes();
        Student student = new Student();
        for (int j = 0; j < childNodes.getLength(); j++) {
            Node childNode = childNodes.item(j);
            if ("name".equals(childNode.getNodeName())) {
                String name = childNode.getTextContent();
                student.setName(name);
                NamedNodeMap nnm = childNode.getAttributes();
                Node n = nnm.item(0);
                student.setSex(n.getTextContent());
            } else if ("nickName".equals(childNode.getNodeName())) {
                String nickName = childNode.getTextContent();
                student.setNickName(nickName);
            }
        }
         list.add(student);
    }
    return list;
}
```

获取天气预报


WebServiceDemo C:\android\Example1\09--Network\WebServiceDemo
.gradle
.idea
app
build
libs
ksoap2-android-assembly-3.3.0-jar-with-dependencies.jar
protobuf-java-3.0.0-beta-1.jar
src
androidTest
main
java
com.example.isszym.webservicedemo
MainActivity
PersonProbuf
TestPb
res
drawable
layout
activity_main.xml
mipmap-hdpi
mipmap-mdpi
mipmap-xhdpi
mipmap-xxhdpi

SOAP格式

- SOAP(Simple Object Access Protocol)是一种信息传递协议，与HTTP协议一起可以用于远程程序调用（RPC），提供WebService。

```
package com.example.isszym.webservicedemo;


import org.ksoap2.SoapEnvelope;
import org.ksoap2.serialization.SoapObject;
import org.ksoap2.serialization.SoapPrimitive;
import org.ksoap2.serialization.SoapSerializationEnvelope;
import org.ksoap2.transport.HttpTransportSE;
import org.w3c.dom.Document;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.io.OutputStreamWriter;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLConnection;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
```

```
public class MainActivity extends AppCompatActivity implements View.OnClickListener {
    private EditText edit_param;
    private TextView txt_result;
    private String result;
    //定义一个Handler用来更新页面:
    private Handler handler = new Handler() {
        public void handleMessage(Message msg) {
            switch (msg.what) {
                case 0x001:
                    txt_result.setText("结果显示: \n" + result);
                    Toast.makeText(MainActivity.this, "获取信息成功", Toast.LENGTH_SHORT).show();
                    break;
                case 0x002:
                    txt_result.setText("结果显示: \n" + result);
                    Toast.makeText(MainActivity.this, "获取信息成功", Toast.LENGTH_SHORT).show();
                    break;
            }
        }
    };
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);      setContentView(R.layout.activity_main);
        edit_param = (EditText) findViewById(R.id.edit_param);
        txt_result = (TextView) findViewById(R.id.txt_result);
        Button btn_query_xml_weather = (Button) findViewById(R.id.btn_query_xml_weather);
        …
        btn_query_xml_weather.setOnClickListener(this);
        btn_query_json_weather.setOnClickListener(this);
        btn_query_soap_weather.setOnClickListener(this);
        btn_post_web_server.setOnClickListener(this);
        btn_protobuf.setOnClickListener(this);
    }
```

```
@Override
public void onClick(View v) {
    final String sss=edit_param.getText().toString();//北京
    switch (v.getId()) {
        case R.id.btn_query_xml_weather:
            new Thread() {
                @Override
                public void run() {
                    try {
                        result = getXmlWeather("北京");
                        handler.sendEmptyMessage(0x001);
                    }catch(Exception ex){
                        ex.printStackTrace();
                    }
                }
            }.start();
            break;
        case R.id.btn_query_json_weather:
            new Thread() {
                @Override
                public void run() {
                    try {
                        result = getJsonWeather("北京");
                        System.out.print(result);
                        handler.sendEmptyMessage(0x001);
                    }catch(Exception ex){
                        ex.printStackTrace();
                    }
                }
            }.start();
            break;
```

```
case R.id.btn_query_soap_weather:
    new Thread() {
        @Override
        public void run() {
            try {
                result = getSoapWeather("北京");
                System.out.print(result);
                handler.sendEmptyMessage(0x001);
            }catch(Exception ex){
                ex.printStackTrace();
            }
        }
    }.start();
    break;

 case R.id.btn_post_web_server:
    new Thread(new Runnable() {
        public void run() {
            try {
                result = postToWebServer();
                System.out.print(result);
                handler.sendEmptyMessage(0x001);
            }catch(Exception ex){
                ex.printStackTrace();
            }
        }
    }).start();
    break;
```

```
            case R.id.btn_protobuf:
                new Thread() {
                    @Override
                    public void run() {
                        try {
                            result=TestPb.getProtoBufString();
                            System.out.print(result);
                            handler.sendEmptyMessage(0x001);
                        }catch(Exception ex){
                            ex.printStackTrace();
                        }
                    }
                }.start();
                break;

        }
    }

    static String getInputStreamText(InputStream is) throws Exception {
        InputStreamReader isr = new InputStreamReader(is, "utf8");
        BufferedReader br = new BufferedReader(isr);
        StringBuilder sb=new StringBuilder();
        String line;
        while ((line = br.readLine()) != null) {
            sb.append(line);
        }
        return sb.toString();
    }
```

```
public String getJsonWeather(String city) throws Exception{
    String city1 = java.net.URLEncoder.encode(city, "utf-8");
    String apiUrl = String.format("https://www.sojson.com/open/api/weather/json.shtml?city=%s",
                                                                        city1);
    URL url= new URL(apiUrl);
    URLConnection open = url.openConnection();
    InputStream inputStream = open.getInputStream();
    return getInputStreamText(inputStream);
}


public String getXmlWeather(String city)  throws Exception{
    String city1 = java.net.URLEncoder.encode(city, "utf-8");
    String apiUrl = String.format("https://www.sojson.com/open/api/weather/xml.shtml?city=%s",
                                    city1);
    URL url= new URL(apiUrl);
    URLConnection open = url.openConnection();
    InputStream inputStream = open.getInputStream();
    return getInputStreamText(inputStream);
}
```

```
    public String getSoapWeather(String city) throws Exception{
        //soap协议  基于http
        String service_url = "http://ws.webxml.com.cn/WebServices/WeatherWS.asmx";
        String name_space = "http://WebXml.com.cn/";
        String method_name = "getWeather";   //具体请求的服务

        HttpTransportSE httpTransportSE = new HttpTransportSE(service_url);          //创建信封
        SoapSerializationEnvelope envelope = new SoapSerializationEnvelope(SoapEnvelope.VER11);
        SoapObject soapObject = new SoapObject(name_space, method_name);
        soapObject.addProperty("theCityCode", city);
        soapObject.addProperty("theUserID", "905f90f0788f41498a29affacc38d27c");
        envelope.bodyOut = soapObject;
        envelope.dotNet = true;   //设置跨语言的兼容性
        httpTransportSE.call(name_space + method_name, envelope);   //连接服务器并发送请求
        if (envelope.getResponse() == null) {//判断响应消息不为空，如果为空就没有请求成功
            return "请求失败！";
        }
       // SoapObject result = (SoapObject) envelope.getResponse();
        SoapObject result1 = (SoapObject) envelope.bodyIn;
        //int count = result.getPropertyCount();   //得到属性集的数量
        SoapObject subSoapObj = (SoapObject) result1.getProperty("getWeatherResult");
        String ss = "属性个数: " + subSoapObj.getPropertyCount();
        for (int i = 0; i < subSoapObj.getPropertyCount(); i++) {
             SoapPrimitive childResult = (SoapPrimitive) subSoapObj.getProperty(i);
            ss=ss+"   " + childResult.getName()+":"+childResult.getValue();
        }
        return ss;
    }
}
```

```
public String postToWebServer() throws Exception {
            HttpURLConnection client = null;
        //      try {
                byte[] data = new String("name="
                            +java.net.URLEncoder.encode("测试", "UTF-8")
                            +"&age=27").getBytes("UTF-8");
                URL url =
                   new URL("http://172.18.187.11:8080/jsp/getFromAndroid.jsp");
                client = (HttpURLConnection) url.openConnection();
                client.setRequestMethod("POST");
                client.setConnectTimeout(1500);
                client.setReadTimeout(1500);
                client.setDoOutput(true);
                client.setDoInput(true);
                OutputStream out = client.getOutputStream();
                out.write(data);
                out.flush();
                out.close();
                //当调用getInputStream方法时才真正将请求体数据上传至服务器
                InputStream is = client.getInputStream();
                Log.d("TAG",client.getResponseCode()+""); //打印响应状态码，200成功
                return getInputStreamText(is);
}
```

ProtoBuf格式

msg.proto

```
option java_package = "com.example.isszym.webservicedemo";
option java_outer_classname = "PersonProbuf";
message Person {
  required string name = 1;
  required int32 id = 2;
  optional string email = 3;
  enum PhoneType {
    MOBILE = 0;
    HOME = 1;
    WORK = 2;
  }
 message PhoneNumber {
    required string number = 1;
    optional PhoneType type = 2 [default = HOME];
  }
  repeated PhoneNumber phone = 4;
```

```
 message CountryInfo {
        required string name = 1;
        required string code = 2;
        optional int32 number = 3;
  }
}
message AddressBook {
  repeated Person person = 1;
}
```

```
C:>protoc --java_out=./src ./msg.proto
C:>java -cp .;./protobuf-java-3.0.0-beta-1.jar TestPb
TestName, kkk@email.com
131111111
011111
[B@65b54208
```

PersonProbuf.class

```
import com.google.protobuf.InvalidProtocolBufferException;
import com.example.isszym.webservicedemo.PersonProbuf;
import com.example.isszym.webservicedemo.PersonProbuf.Person;
import com.example.isszym.webservicedemo.PersonProbuf.Person.PhoneNumber;

public class TestPb {
   public static String getProtoBufString() {
      // TODO Auto-generated method stub
      PersonProbuf.Person.Builder builder = PersonProbuf.Person.newBuilder();
      builder.setEmail("kkk@email.com");
      builder.setId(1);
      builder.setName("TestName");
      builder.addPhone(PersonProbuf.Person.PhoneNumber.newBuilder()
                  .setNumber("131111111").setType(PersonProbuf.Person.PhoneType.MOBILE));
      builder.addPhone(PersonProbuf.Person.PhoneNumber.newBuilder()
                  .setNumber("011111").setType(PersonProbuf.Person.PhoneType.HOME));

      Person person = builder.build();
      byte[] buf = person.toByteArray();
        String res="";
```

```
        try {
            Person person2 = PersonProbuf.Person.parseFrom(buf);
            //System.out.println(person2.getName() + ", " + person2.getEmail());
            res=res+" name:"+person2.getName() + ", email:" + person2.getEmail();
            List<PhoneNumber> lstPhones = person2.getPhoneList();
            for (PhoneNumber phoneNumber : lstPhones) {
            // System.out.println(phoneNumber.getNumber());
                res=res+" phone#:"+phoneNumber.getNumber();
            }
        } catch (InvalidProtocolBufferException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
          return res;
        //System.out.println(buf);

    }

}
```