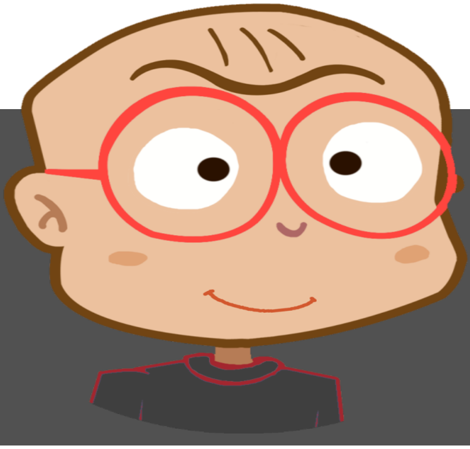


Parse XML, JSON

START »

Learning Goals

After the course, attendees will be able to:

- Understanding about structure of JSON and XML data
- Know about how to parse both of two data

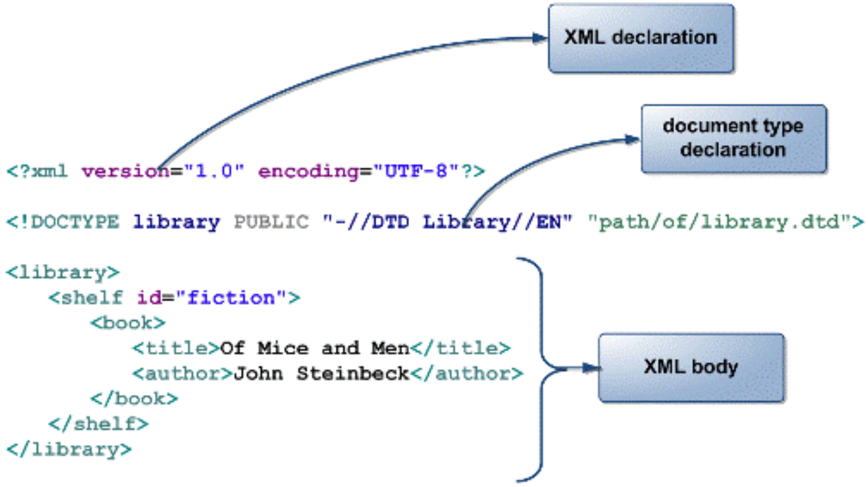
Agenda

1. What is XML?
2. How to parse XML?
3. What is JSON?
4. How to parse JSON?
5. Practices

What is XML

What is XML?

- XML stands for Extensible Markup Language.XML is a very popular format and commonly used for sharing data on the internet
- Websites that frequently update their content, such as news sites or blogs, often provide an XML feed so that external programs can keep abreast of content changes



```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE library PUBLIC "-//DTD Library//EN" "path/of/library.dtd">

<library>
   <shelf id="fiction">
      <book>
         <title>Of Mice and Men</title>
         <author>John Steinbeck</author>
      </book>
   </shelf>
</library>
```

Common ways for parse XML:

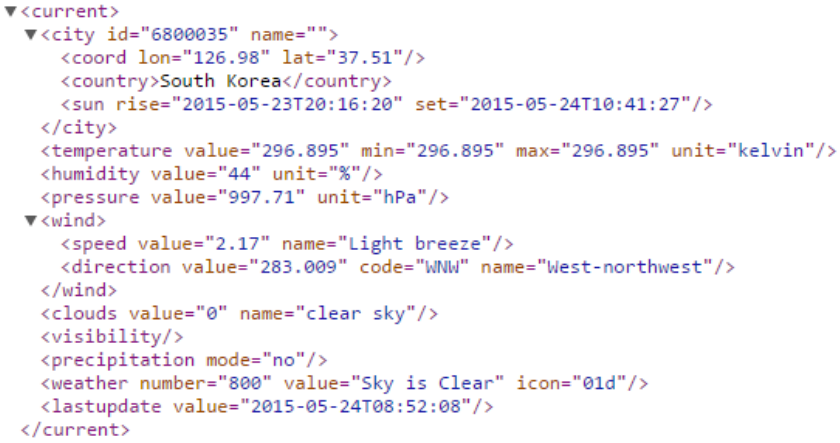
- Android provides three types of XML parsers which are **DOM,SAX and XMLPullParser**.
- Among all of them android recommend XMLPullParser because it is efficient and easy to use.

Refer compare: Comparing methods of XML parsing in Android

How to parse XML

Analyze the XML data

- The first step in parsing a feed is to decide which fields you're interested in. The parser extracts data for those fields and ignores the rest.

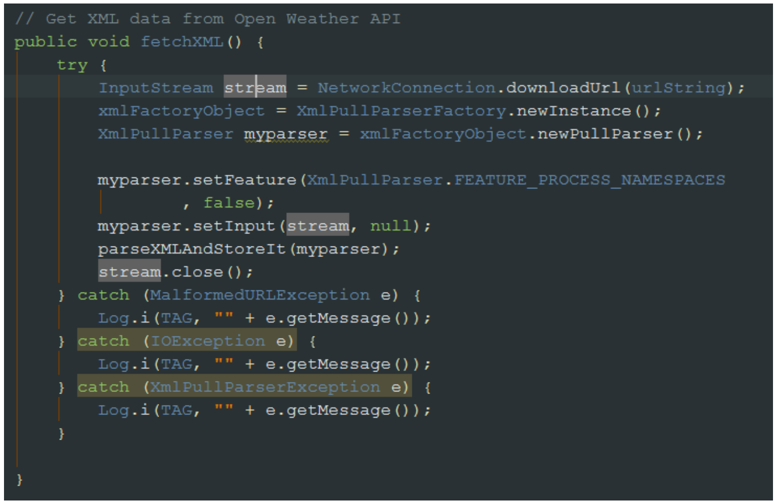


```
▼<current>
  ▼<city id="6800035" name="">
     <coord lon="126.98" lat="37.51"/>
     <country>South Korea</country>
     <sun rise="2015-05-23T20:16:20" set="2015-05-24T10:41:27"/>
   </city>
   <temperature value="296.895" min="296.895" max="296.895" unit="kelvin"/>
   <humidity value="44" unit="%"/>
   <pressure value="997.71" unit="hPa"/>
  ▼<wind>
     <speed value="2.17" name="Light breeze"/>
     <direction value="283.009" code="WNW" name="West-northwest"/>
   </wind>
   <clouds value="0" name="clear sky"/>
   <visibility/>
   <precipitation mode="no"/>
   <weather number="800" value="Sky is Clear" icon="01d"/>
   <lastupdate value="2015-05-24T08:52:08"/>
 </current>
```

Refer : Weather API XML

Instantiate the Parse

The next step is to instantiate a parser and kick off the parsing process. In this snippet, a parser is initialized to not process namespaces, and to use the provided InputStream as its input

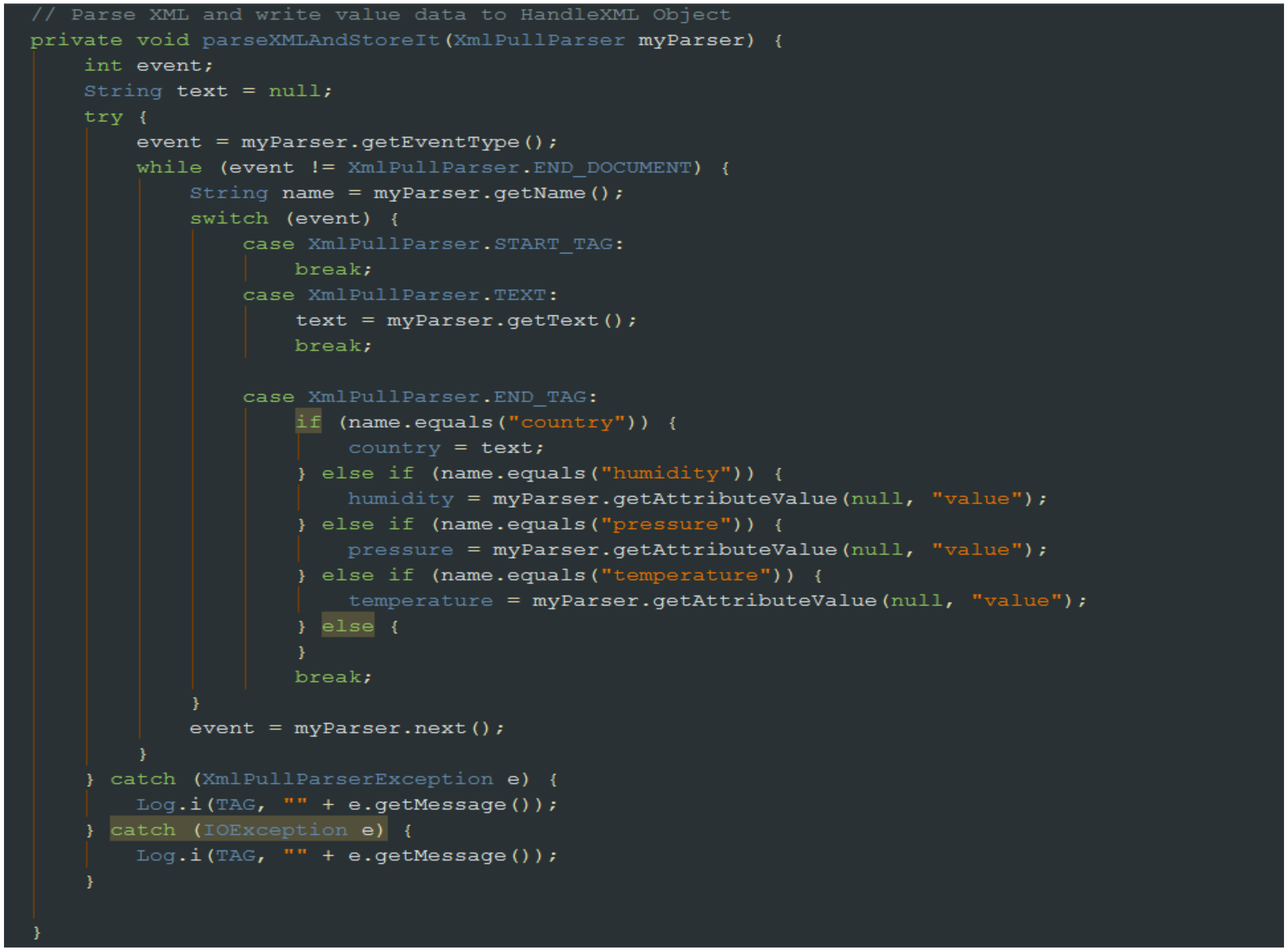


```java
// Get XML data from Open Weather API
public void fetchXML() {
    try {
        InputStream stream = NetworkConnection.downloadUrl(urlString);
        xmlFactoryObject = XmlPullParserFactory.newInstance();
        XmlPullParser myparser = xmlFactoryObject.newPullParser();

        myparser.setFeature(XmlPullParser.FEATURE_PROCESS_NAMESPACES
                , false);
        myparser.setInput(stream, null);
        parseXMLAndStoreIt(myparser);
        stream.close();
    } catch (MalformedURLException e) {
        Log.i(TAG, "" + e.getMessage());
    } catch (IOException e) {
        Log.i(TAG, "" + e.getMessage());
    } catch (XmlPullParserException e) {
        Log.i(TAG, "" + e.getMessage());
    }

}
```

Parse XML



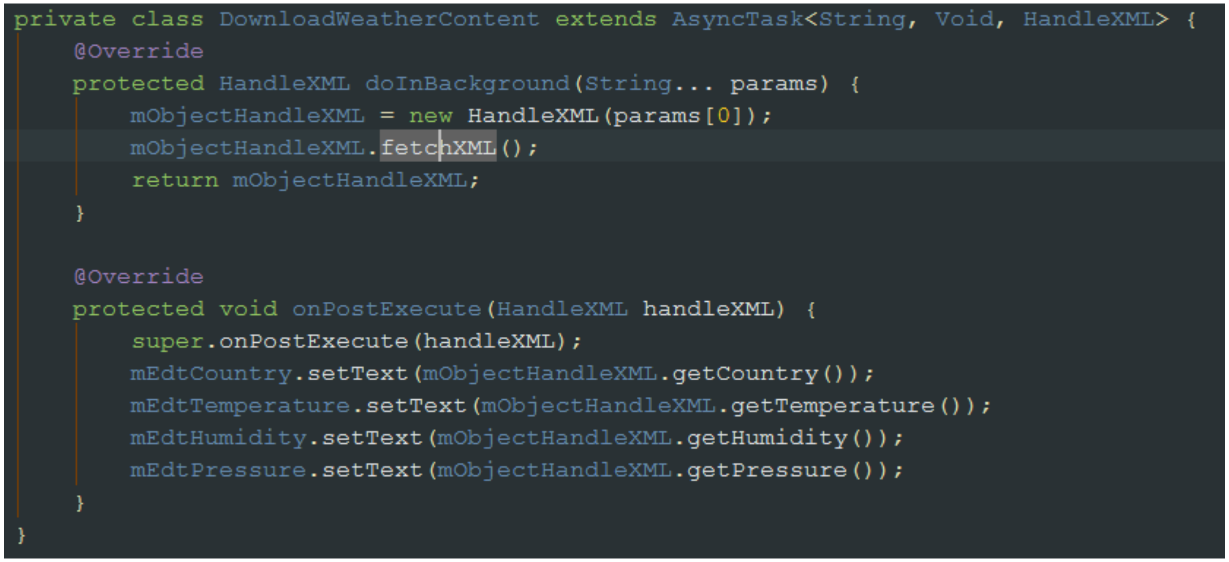
```java
// Parse XML and write value data to HandleXML Object
private void parseXMLAndStoreIt(XmlPullParser myParser) {
    int event;
    String text = null;
    try {
        event = myParser.getEventType();
        while (event != XmlPullParser.END_DOCUMENT) {
            String name = myParser.getName();
            switch (event) {
                case XmlPullParser.START_TAG:
                    break;
                case XmlPullParser.TEXT:
                    text = myParser.getText();
                    break;

                case XmlPullParser.END_TAG:
                    if (name.equals("country")) {
                        country = text;
                    } else if (name.equals("humidity")) {
                        humidity = myParser.getAttributeValue(null, "value");
                    } else if (name.equals("pressure")) {
                        pressure = myParser.getAttributeValue(null, "value");
                    } else if (name.equals("temperature")) {
                        temperature = myParser.getAttributeValue(null, "value");
                    } else {
                    }
                    break;
            }
            event = myParser.next();
        }
    } catch (XmlPullParserException e) {
        Log.i(TAG, "" + e.getMessage());
    } catch (IOException e) {
        Log.i(TAG, "" + e.getMessage());
    }

}
```

Consume XML Data

The example application fetches and parses the XML feed within an AsyncTask. This takes the processing off the main UI thread. When processing is complete, the app updates the UI in the main activity.



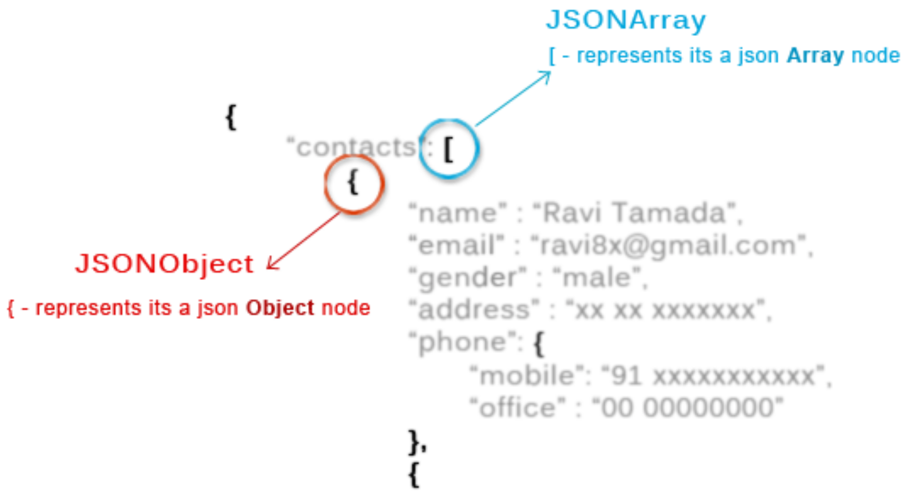
```
private class DownloadWeatherContent extends AsyncTask<String, Void, HandleXML> {
    @Override
    protected HandleXML doInBackground(String... params) {
        mObjectHandleXML = new HandleXML(params[0]);
        mObjectHandleXML.fetchXML();
        return mObjectHandleXML;
    }

    @Override
    protected void onPostExecute(HandleXML handleXML) {
        super.onPostExecute(handleXML);
        mEdtCountry.setText(mObjectHandleXML.getCountry());
        mEdtTemperature.setText(mObjectHandleXML.getTemperature());
        mEdtHumidity.setText(mObjectHandleXML.getHumidity());
        mEdtPressure.setText(mObjectHandleXML.getPressure());
    }
}
```

What is JSON

What is JSON?

- JSON stands for JavaScript Object Notation
- JSON is very light weight, structured, easy to parse and much human readable.
- JSON is best alternative to XML when your android app needs to interchange data with your server


JSONArray
[- represents its a json **Array** node
JSONObject
{ - represents its a json **Object** node

```
{
    "contacts": [
        {
                "name" : "Ravi Tamada",
                "email" : "ravi8x@gmail.com",
                "gender" : "male",
                "address" : "xx xx xxxxxxx",
                "phone": {
                        "mobile": "91 xxxxxxxxxx",
                        "office" : "00 00000000"
                },
                {
```

The difference between [and {

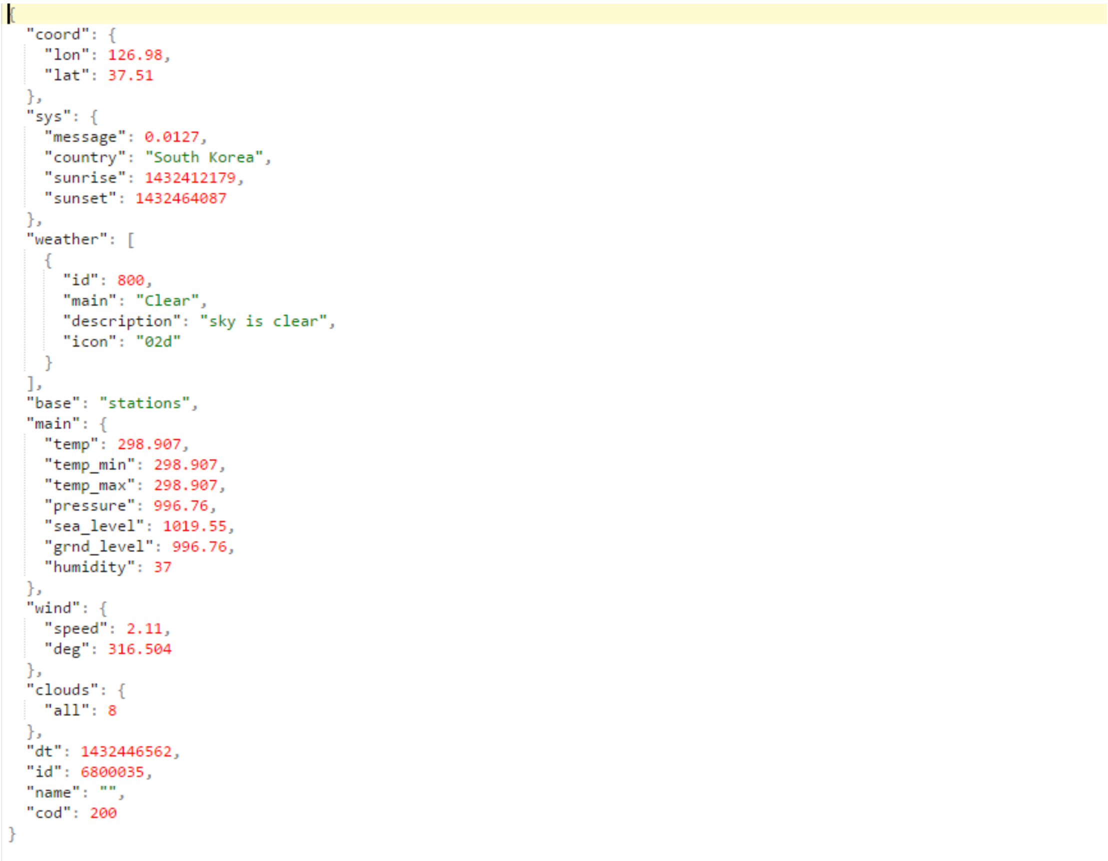
- The difference between [and { is, the square bracket ([) represents starting of an **JSONArray** node whereas curly bracket ({) represents **JSONObject**.
- If your JSON node starts with **[**, then we should use **getJSONArray()** method
- Same as if the node starts with **{**, then we should use **getJSONObject()** method.

How to parse JSON

Analyze the JSONdata

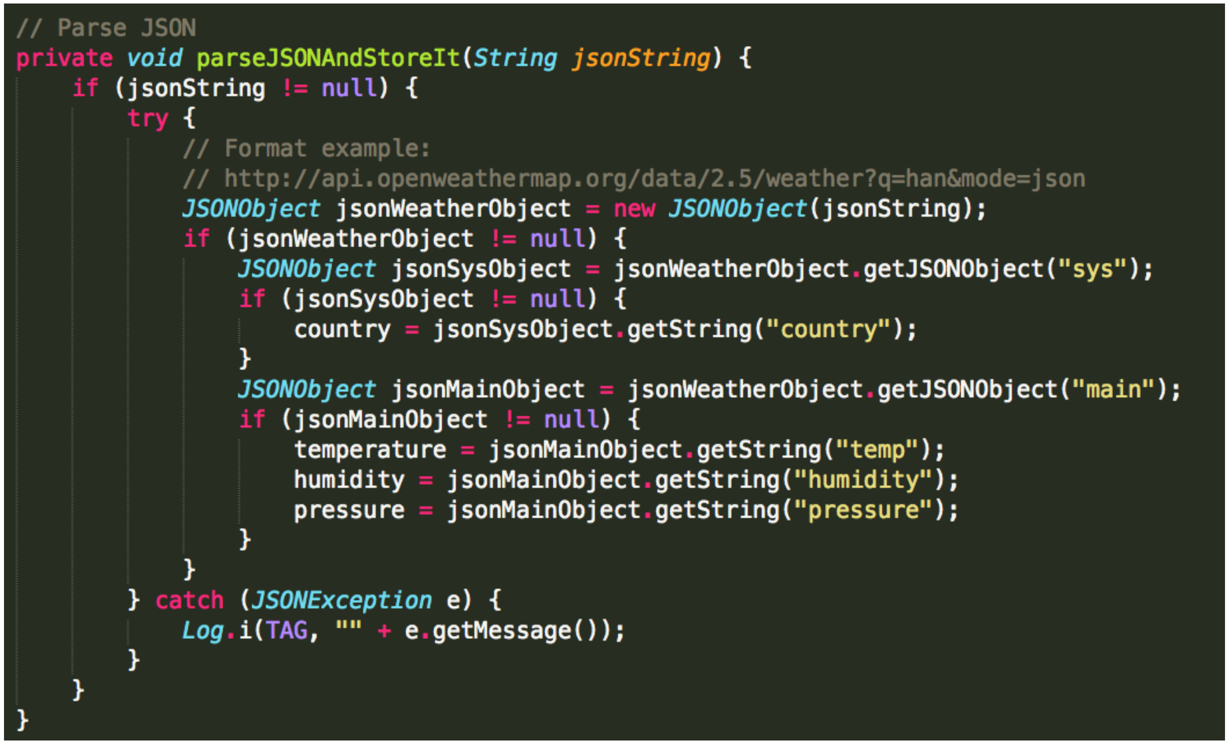
- Almost step will the same as parse XML. Only have different in analyze data and Parse.

Refer :
Weather API JSON



```
{
  "coord": {
    "lon": 126.98,
    "lat": 37.51
  },
  "sys": {
    "message": 0.0127,
    "country": "South Korea",
    "sunrise": 1432412179,
    "sunset": 143264087
  },
  "weather": [
    {
      "id": 800,
      "main": "Clear",
      "description": "sky is clear",
      "icon": "02d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 298.907,
    "temp_min": 298.907,
    "temp_max": 298.907,
    "pressure": 996.76,
    "sea_level": 1019.55,
    "grnd_level": 996.76,
    "humidity": 37
  },
  "wind": {
    "speed": 2.11,
    "deg": 316.504
  },
  "clouds": {
    "all": 8
  },
  "dt": 1432446562,
  "id": 6800035,
  "name": "",
  "cod": 200
}
```

Parse JSON



```
// Parse JSON
private void parseJSONAndStoreIt(String jsonString) {
    if (jsonString != null) {
        try {
            // Format example:
            // http://api.openweathermap.org/data/2.5/weather?q=han&mode=json
            JSONObject jsonWeatherObject = new JSONObject(jsonString);
            if (jsonWeatherObject != null) {
                JSONObject jsonSysObject = jsonWeatherObject.getJSONObject("sys");
                if (jsonSysObject != null) {
                    country = jsonSysObject.getString("country");
                }
                JSONObject jsonMainObject = jsonWeatherObject.getJSONObject("main");
                if (jsonMainObject != null) {
                    temperature = jsonMainObject.getString("temp");
                    humidity = jsonMainObject.getString("humidity");
                    pressure = jsonMainObject.getString("pressure");
                }
            }
        } catch (JSONException e) {
            Log.i(TAG, "" + e.getMessage());
        }
    }
}
```

Exit Course

THANK YOU