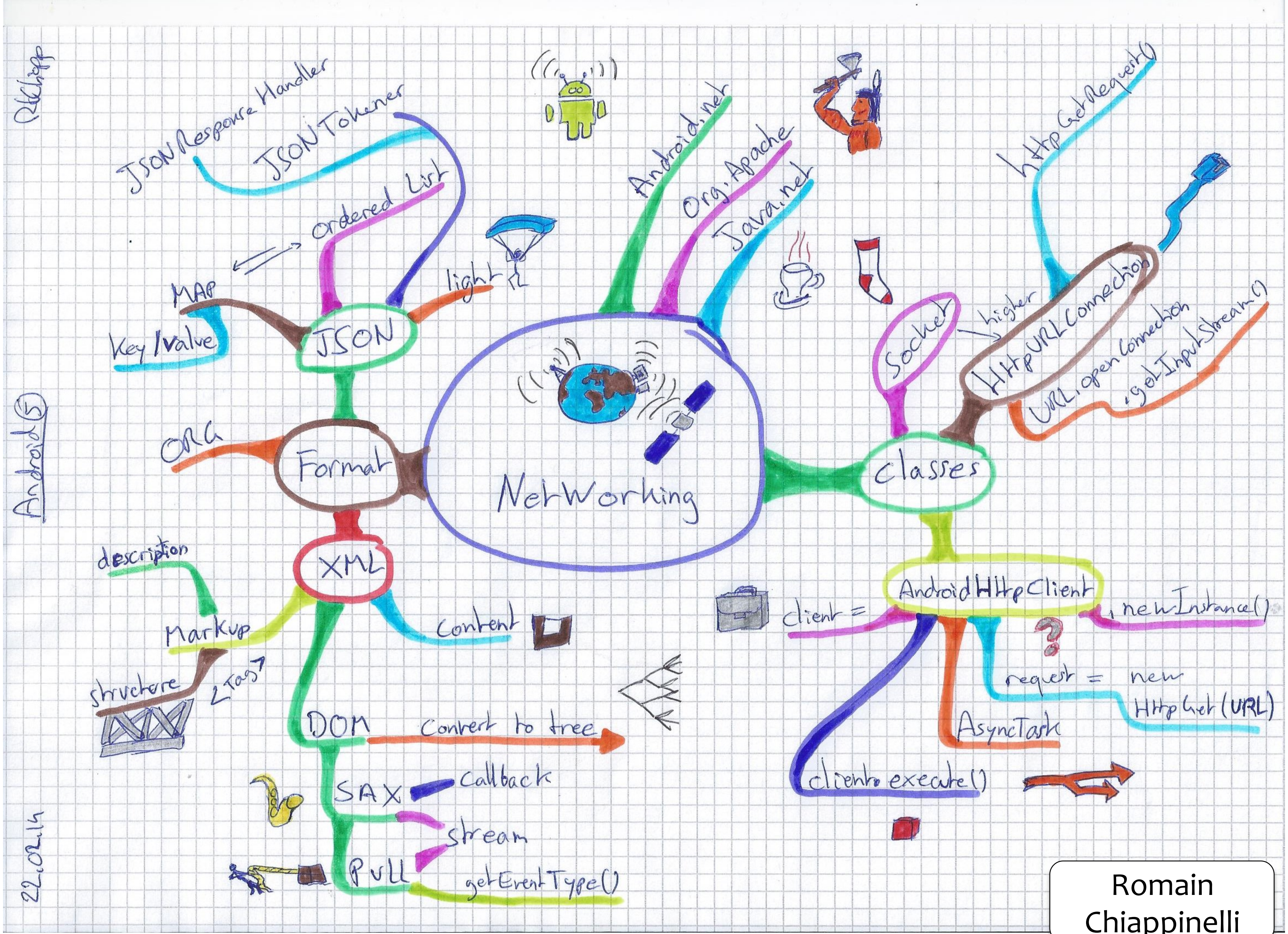


Romain Chiappinelli

Networking

Today's Topics

Networking

Android networking Classes

Processing HTTP Responses

Networking

Early handheld devices gave us mobility

But with limited connectivity

Today's devices have greater mobility and connectivity

Many applications use data and services via the Internet

Networking

Android includes multiple networking support classes, e.g.,

java.net – (Socket, URL)

org.apache – (HttpRequest, HttpResponse)

android.net – (URI, AndroidHttpClient, AudioStream)

Example Application

Application sends a request to a networked server for earthquake data

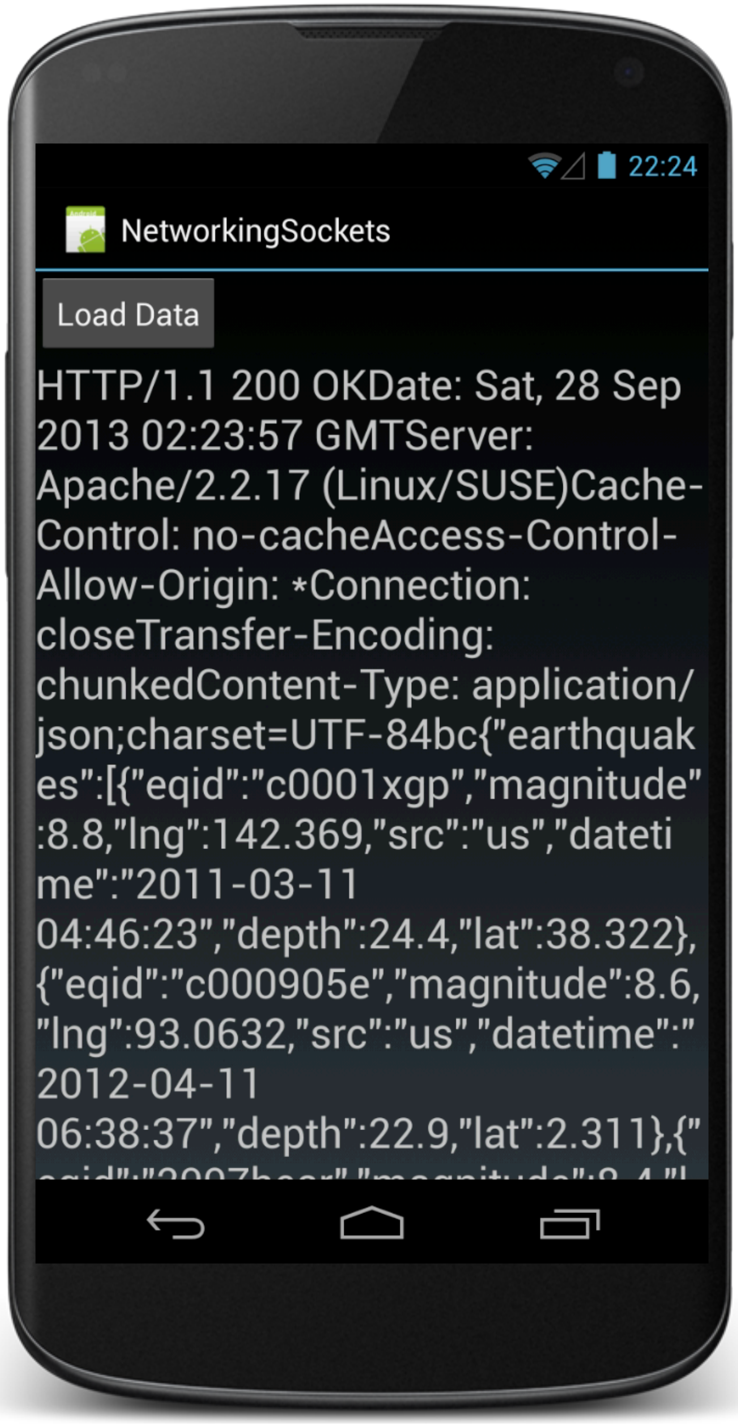
Then Displays the requested data

Sending HTTP Requests

Socket

HttpURLConnection

AndroidHttpClient



```
HTTP/1.1 200 OKDate: Sat, 28 Sep
2013 02:23:57 GMTServer:
Apache/2.2.17 (Linux/SUSE)Cache-
Control: no-cacheAccess-Control-
Allow-Origin: *Connection:
closeTransfer-Encoding:
chunkedContent-Type: application/
json;charset=UTF-84bc{"earthquak
es":[{"eqid":"c0001xgp","magnitude"
:8.8,"lng":142.369,"src":"us","dateti
me":"2011-03-11
04:46:23","depth":24.4,"lat":38.322},
{"eqid":"c000905e","magnitude":8.6,
"lng":93.0632,"src":"us","datetime":"
2012-04-11
06:38:37","depth":22.9,"lat":2.311},{"
```

NetworkingSockets

```java
public class NetworkingSocketsActivity extends Activity {
    TextView mTextView;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        mTextView = (TextView) findViewById(R.id.textView1);

        final Button loadButton = (Button) findViewById(R.id.button1);
        loadButton.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View v) {
                new HttpGetTask().execute();

            }
        });
    }
```

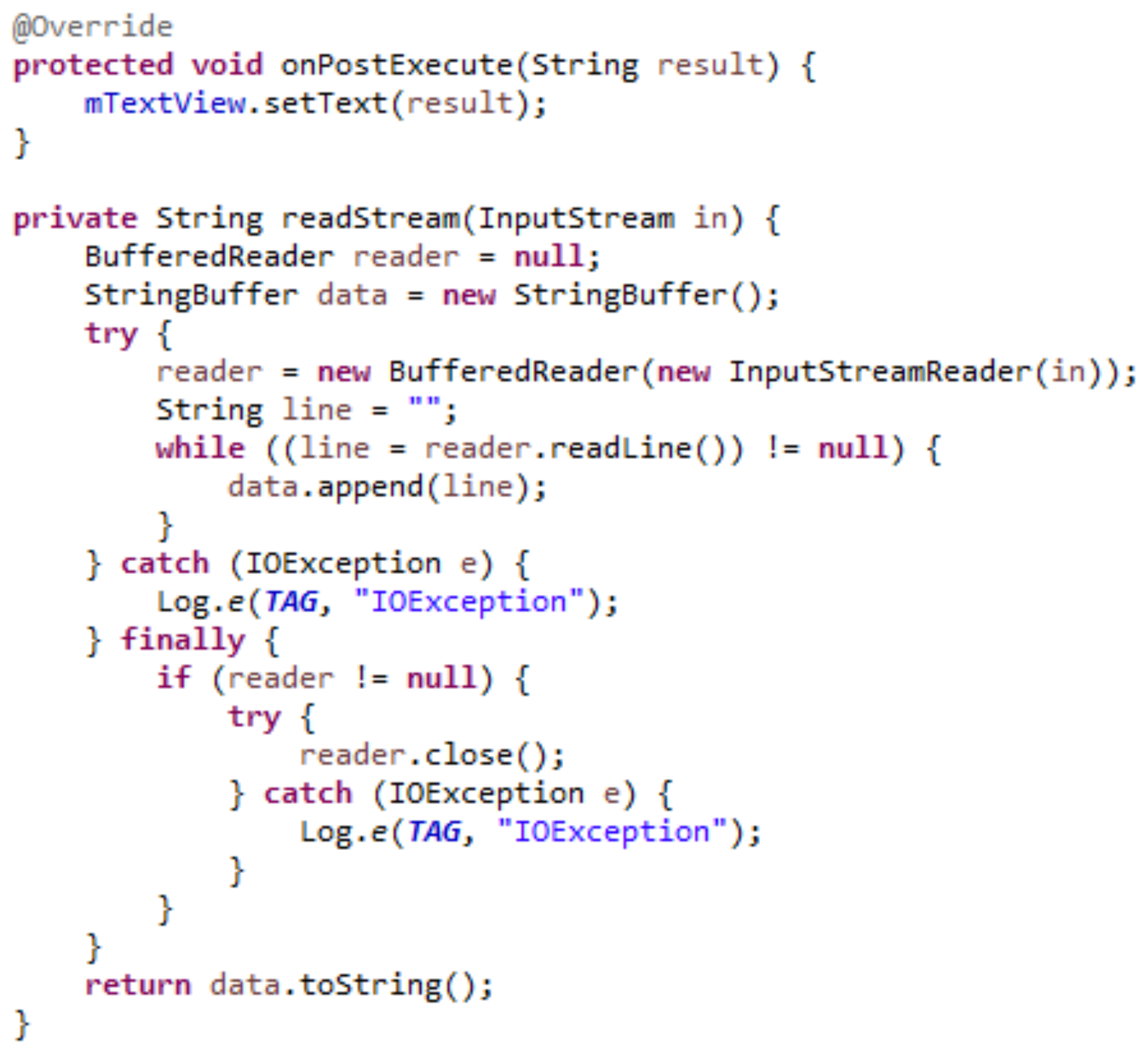
NetworkingSockets

```java
private class HttpGetTask extends AsyncTask<Void, Void, String> {

    private static final String HOST = "api.geonames.org";

    // Get your own user name at http://www.geonames.org/login
    private static final String USER_NAME = "aporter";
    private static final String HTTP_GET_COMMAND = "GET /earthquakesJSON?north=44.1&south=-9.9&east=-22.4&west=55.2&username="
            + USER_NAME
            + " HTTP/1.1"
            + "\n"
            + "Host: "
            + HOST
            + "\n"
            + "Connection: close" + "\n\n";

    private static final String TAG = "HttpGet";
```

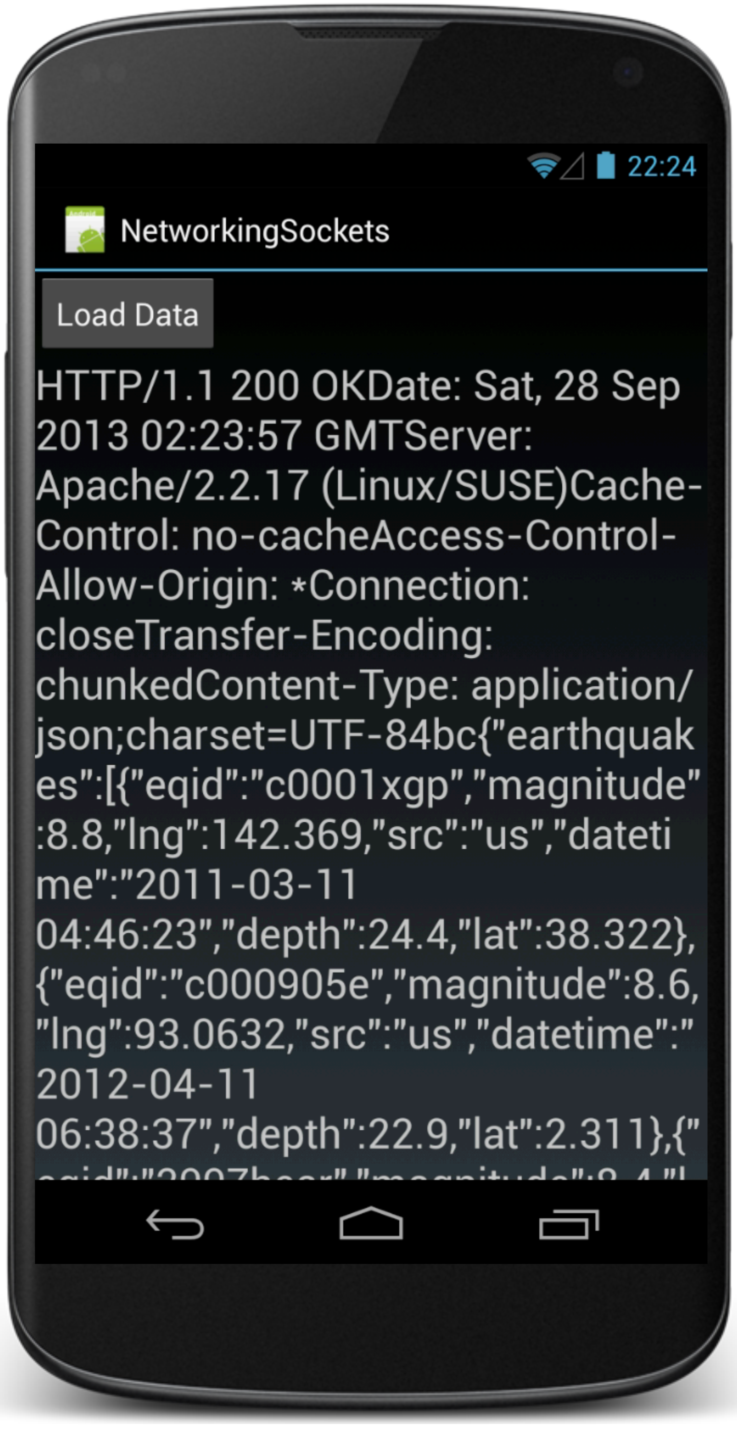
NetworkingSockets

```
@Override
protected String doInBackground(Void... params) {
    Socket socket = null;
    String data = "";

    try {
        socket = new Socket(HOST, 80);
        PrintWriter pw = new PrintWriter(new OutputStreamWriter(
                socket.getOutputStream()), true);
        pw.println(HTTP_GET_COMMAND);

        data = readStream(socket.getInputStream());

    } catch (UnknownHostException exception) {
        exception.printStackTrace();
    } catch (IOException exception) {
        exception.printStackTrace();
    } finally {
        if (null != socket)
            try {
                socket.close();
            } catch (IOException e) {
                Log.e(TAG, "IOException");
            }
    }
    return data;
}
```

NetworkingSockets



```
@Override
protected void onPostExecute(String result) {
    mTextView.setText(result);
}

private String readStream(InputStream in) {
    BufferedReader reader = null;
    StringBuffer data = new StringBuffer();
    try {
        reader = new BufferedReader(new InputStreamReader(in));
        String line = "";
        while ((line = reader.readLine()) != null) {
            data.append(line);
        }
    } catch (IOException e) {
        Log.e(TAG, "IOException");
    } finally {
        if (reader != null) {
            try {
                reader.close();
            } catch (IOException e) {
                Log.e(TAG, "IOException");
            }
        }
    }
    return data.toString();
}
```

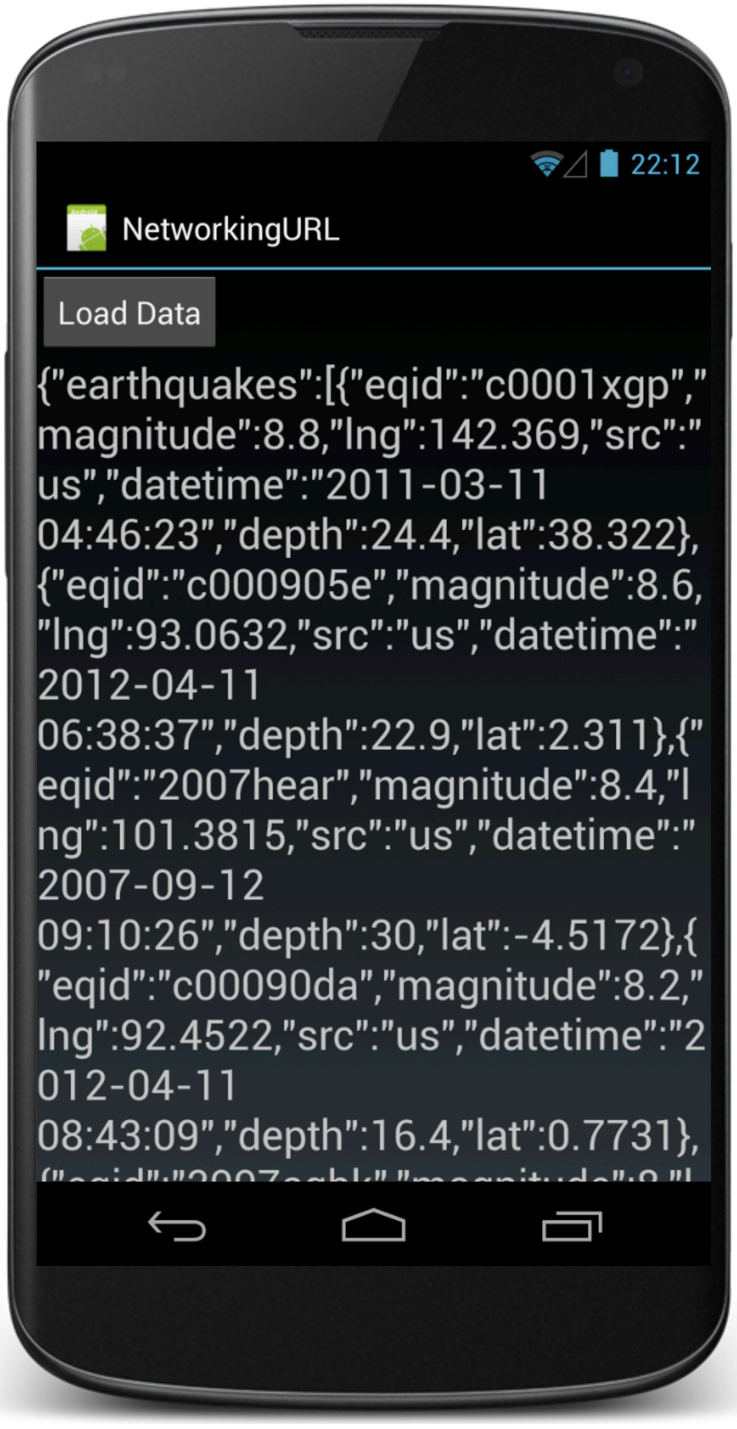


```
HTTP/1.1 200 OKDate: Sat, 28 Sep
2013 02:23:57 GMTServer:
Apache/2.2.17 (Linux/SUSE)Cache-
Control: no-cacheAccess-Control-
Allow-Origin: *Connection:
closeTransfer-Encoding:
chunkedContent-Type: application/
json;charset=UTF-84bc{"earthquak
es":[{"eqid":"c0001xgp","magnitude"
:8.8,"lng":142.369,"src":"us","dateti
me":"2011-03-11
04:46:23","depth":24.4,"lat":38.322},
{"eqid":"c000905e","magnitude":8.6,
"lng":93.0632,"src":"us","datetime":"
2012-04-11
06:38:37","depth":22.9,"lat":2.311},{"
```

HttpURLConnection

Higher-level than Sockets

Less Flexible API than HttpAndroidClient



```
{"earthquakes":[{"eqid":"c0001xgp","
magnitude":8.8,"lng":142.369,"src":"
us","datetime":"2011-03-11
04:46:23","depth":24.4,"lat":38.322},
{"eqid":"c000905e","magnitude":8.6,
"lng":93.0632,"src":"us","datetime":"
2012-04-11
06:38:37","depth":22.9,"lat":2.311},{"
eqid":"2007hear","magnitude":8.4,"l
ng":101.3815,"src":"us","datetime":"
2007-09-12
09:10:26","depth":30,"lat":-4.5172},{
"eqid":"c00090da","magnitude":8.2,"
lng":92.4522,"src":"us","datetime":"2
012-04-11
08:43:09","depth":16.4,"lat":0.7731},
```

NetworkingURL

```
public class NetworkingURLActivity extends Activity {
    private TextView mTextView;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.main);
        mTextView = (TextView) findViewById(R.id.textView1);

        final Button loadButton = (Button) findViewById(R.id.button1);
        loadButton.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View v) {
                new HttpGetTask().execute();
            }
        });
    }
```

NetworkingURL

```
private class HttpGetTask extends AsyncTask<Void, Void, String> {

    private static final String TAG = "HttpGetTask";

    // Get your own user name at http://www.geonames.org/login
    private static final String USER_NAME = "aporter";
    private static final String URL = "http://api.geonames.org/earthquakesJSON?north=44.1&south=-9.9&east=-22.4&west=55.2&username="
            + USER_NAME;

    @Override
    protected String doInBackground(Void... params) {
        String data = "";
        HttpURLConnection httpUrlConnection = null;

        try {
            httpUrlConnection = (HttpURLConnection) new URL(URL)
                    .openConnection();

            InputStream in = new BufferedInputStream(
                    httpUrlConnection.getInputStream());

            data = readStream(in);

        } catch (MalformedURLException exception) {
            Log.e(TAG, "MalformedURLException");
        } catch (IOException exception) {
            Log.e(TAG, "IOException");
        } finally {
            if (null != httpUrlConnection)
                httpUrlConnection.disconnect();
        }
        return data;
    }
```

NetworkingURL

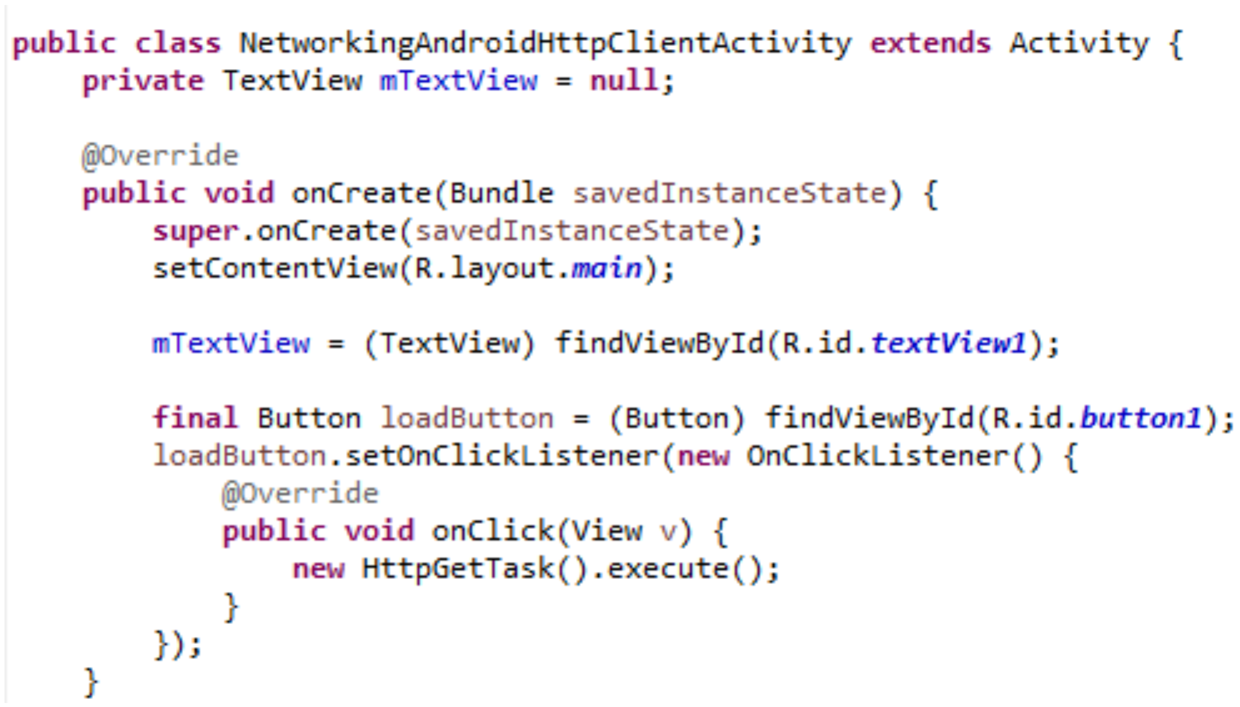
```java
@Override
protected void onPostExecute(String result) {
    mTextView.setText(result);
}

private String readStream(InputStream in) {
    BufferedReader reader = null;
    StringBuffer data = new StringBuffer("");
    try {
        reader = new BufferedReader(new InputStreamReader(in));
        String line = "";
        while ((line = reader.readLine()) != null) {
            data.append(line);
        }
    } catch (IOException e) {
        Log.e(TAG, "IOException");
    } finally {
        if (reader != null) {
            try {
                reader.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
    return data.toString();
}
```

AndroidHttpClient

An implementation of Apache's DefaultHttpClient

Breaks HTTP Transaction into separate Request and Response Objects

NetworkingAndroidHttpClient



```java
public class NetworkingAndroidHttpClientActivity extends Activity {
    private TextView mTextView = null;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        mTextView = (TextView) findViewById(R.id.textView1);

        final Button loadButton = (Button) findViewById(R.id.button1);
        loadButton.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                new HttpGetTask().execute();
            }
        });
    }
```

NetworkingAndroidHttpClient

```
private class HttpGetTask extends AsyncTask<Void, Void, String> {

    // Get your own user name at http://www.geonames.org/login
    private static final String USER_NAME = "aporter";

    private static final String URL = "http://api.geonames.org/earthquakesJSON?north=44.1&south=-9.9&east=-22.4&west=55.2&username="
            + USER_NAME;

    AndroidHttpClient mClient = AndroidHttpClient.newInstance("");

    @Override
    protected String doInBackground(Void... params) {

        HttpGet request = new HttpGet(URL);
        ResponseHandler<String> responseHandler = new BasicResponseHandler();

        try {

            return mClient.execute(request, responseHandler);

        } catch (ClientProtocolException exception) {
            exception.printStackTrace();
        } catch (IOException exception) {
            exception.printStackTrace();
        }
        return null;
    }

    @Override
    protected void onPostExecute(String result) {

        if (null != mClient)
            mClient.close();

        mTextView.setText(result);

    }
}
```

Processing Http Responses

Several popular formats including

JSON

XML

Javascript Object Notation (JSON)

Intended to be a lightweight data interchange format

Data packaged in two types of structures:

Maps of key/value pairs

Ordered lists of values

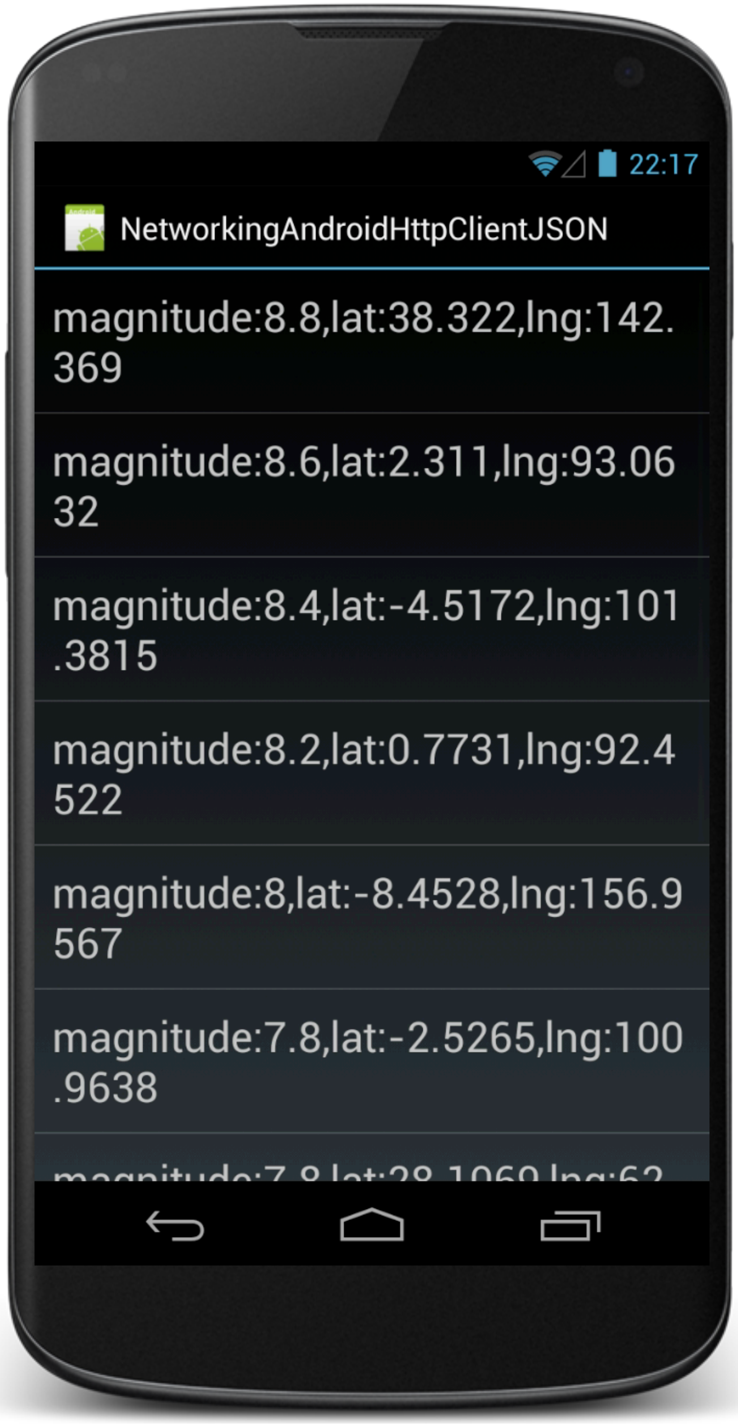
See: http://www.json.org/

Earthquake Data (JSON Output)

http://api.geonames.org/earthquakesJSON?north=44.1&south=-9.9&east=-22.4&west=55.2&username=demo

Earthquake Data (JSON Output)

```
{"earthquakes”: [
 {"eqid":"c0001xgp","magnitude":8.8,"lng":142.369,
  "src":"us”, "datetime":"2011-03-11 04:46:23","depth":
   24.4,"lat":38.322}
 {"eqid":"2007hear","magnitude":8.4,"lng":101.3815,
  "src":"us”,"datetime":"2007-09-12 09:10:26","depth":
  30,"lat":-4.5172},
 ...
  {"eqid":"2010xkbv","magnitude":7.5,"lng":91.9379,
   "src":"us”,"datetime":"2010-06-12 17:26:50","depth":
   35,"lat":7.7477}
 ]
}
```



```
magnitude:8.8,lat:38.322,lng:142.
369
magnitude:8.6,lat:2.311,lng:93.06
32
magnitude:8.4,lat:-4.5172,lng:101
.3815
magnitude:8.2,lat:0.7731,lng:92.4
522
magnitude:8,lat:-8.4528,lng:156.9
567
magnitude:7.8,lat:-2.5265,lng:100
.9638
```

NetworkingAndroidHttpClientJSON

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    new HttpGetTask().execute();
}

private class HttpGetTask extends AsyncTask<Void, Void, List<String>> {

    // Get your own user name at http://www.geonames.org/login
    private static final String USER_NAME = "aporter";

    private static final String URL = "http://api.geonames.org/earthquakesJSON?north=44.1&south=-9.9&east=-22.4&west=55.2&username="
            + USER_NAME;

    AndroidHttpClient mClient = AndroidHttpClient.newInstance("");

    @Override
    protected List<String> doInBackground(Void... params) {
        HttpGet request = new HttpGet(URL);
        JSONResponseHandler responseHandler = new JSONResponseHandler();
        try {
            return mClient.execute(request, responseHandler);
        } catch (ClientProtocolException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }

    @Override
    protected void onPostExecute(List<String> result) {
        if (null != mClient)
            mClient.close();
        setListAdapter(new ArrayAdapter<String>(
                NetworkingAndroidHttpClientJSONActivity.this,
                R.layout.list_item, result));
    }
}
```

NetworkingAndroidHttpClientJSON

```java
private class JSONResponseHandler implements ResponseHandler<List<String>> {

    private static final String LONGITUDE_TAG = "lng";
    private static final String LATITUDE_TAG = "lat";
    private static final String MAGNITUDE_TAG = "magnitude";
    private static final String EARTHQUAKE_TAG = "earthquakes";

    @Override
    public List<String> handleResponse(HttpResponse response)
            throws ClientProtocolException, IOException {
        List<String> result = new ArrayList<String>();
        String JSONResponse = new BasicResponseHandler()
                .handleResponse(response);
        try {

            // Get top-level JSON Object - a Map
            JSONObject responseObject = (JSONObject) new JSONTokener(
                    JSONResponse).nextValue();

            // Extract value of "earthquakes" key -- a List
            JSONArray earthquakes = responseObject
                    .getJSONArray(EARTHQUAKE_TAG);

            // Iterate over earthquakes list
            for (int idx = 0; idx < earthquakes.length(); idx++) {

                // Get single earthquake data - a Map
                JSONObject earthquake = (JSONObject) earthquakes.get(idx);

                // Summarize earthquake data as a string and add it to
                // result
                result.add(MAGNITUDE_TAG + ":"
                        + earthquake.get(MAGNITUDE_TAG) + ","
                        + LATITUDE_TAG + ":"
                        + earthquake.getString(LATITUDE_TAG) + ","
                        + LONGITUDE_TAG + ":"
                        + earthquake.get(LONGITUDE_TAG));
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
        return result;
    }
}
```

eXtensible Markup Language (XML)

XML documents can contain markup & content

Markup encodes a description of the document's storage layout and logical structure

Content is everything else

See http://www.w3.org/TR/xml

Earthquake Data (XML)

http://api.geonames.org/earthquakes?north=44.1&south=-9.9&east=-22.4&west=55.2& username=demo

Earthquake Data (XML)

```
<geonames>
 <earthquake>
  <src>us</src>
  <eqid>c0001xgp</eqid>
  <datetime>2011-03-11 04:46:23</datetime>
  <lat>38.322</lat>
  <lng>142.369</lng>
  <magnitude>8.8</magnitude>
  <depth>24.4</depth>
 </earthquake>
...
</geonames>
```

Parsing XML

Several types of parsers available

DOM – Converts document into a tree of nodes

SAX – streaming with application callbacks

Pull – Application iterates over XML entries

NetworkingAndroidHttpClientXML

```
class XMLResponseHandler implements ResponseHandler<List<String>> {

    private static final String MAGNITUDE_TAG = "magnitude";
    private static final String LONGITUDE_TAG = "lng";
    private static final String LATITUDE_TAG = "lat";
    private String mLat, mLng, mMag;
    private boolean mIsParsingLat, mIsParsingLng, mIsParsingMag;
    private final List<String> mResults = new ArrayList<String>();

    @Override
    public List<String> handleResponse(HttpResponse response)
            throws ClientProtocolException, IOException {
        try {

            // Create the Pull Parser
            XmlPullParserFactory factory = XmlPullParserFactory.newInstance();
            XmlPullParser xpp = factory.newPullParser();

            // Set the Parser's input to be the XML document in the HTTP Response
            xpp.setInput(new InputStreamReader(response.getEntity()
                    .getContent()));

            // Get the first Parser event and start iterating over the XML document
            int eventType = xpp.getEventType();

            while (eventType != XmlPullParser.END_DOCUMENT) {

                if (eventType == XmlPullParser.START_TAG) {
                    startTag(xpp.getName());
                } else if (eventType == XmlPullParser.END_TAG) {
                    endTag(xpp.getName());
                } else if (eventType == XmlPullParser.TEXT) {
                    text(xpp.getText());
                }
                eventType = xpp.next();
            }
            return mResults;
        } catch (XmlPullParserException e) {
        }
        return null;
    }
```

NetworkingAndroidHttpClientXML

```java
public void startTag(String localName) {
    if (localName.equals(LATITUDE_TAG)) {
        mIsParsingLat = true;
    } else if (localName.equals(LONGITUDE_TAG)) {
        mIsParsingLng = true;
    } else if (localName.equals(MAGNITUDE_TAG)) {
        mIsParsingMag = true;
    }
}

public void text(String text) {
    if (mIsParsingLat) {
        mLat = text.trim();
    } else if (mIsParsingLng) {
        mLng = text.trim();
    } else if (mIsParsingMag) {
        mMag = text.trim();
    }
}

public void endTag(String localName) {
    if (localName.equals(LATITUDE_TAG)) {
        mIsParsingLat = false;
    } else if (localName.equals(LONGITUDE_TAG)) {
        mIsParsingLng = false;
    } else if (localName.equals(MAGNITUDE_TAG)) {
        mIsParsingMag = false;
    } else if (localName.equals("earthquake")) {
        mResults.add(MAGNITUDE_TAG + ":" + mMag + "," + LATITUDE_TAG + ":"
                + mLat + "," + LONGITUDE_TAG + ":" + mLng);
        mLat = null;
        mLng = null;
        mMag = null;
    }
}
```

Next Time

Graphics & Animation

Extra - Networking Permissions

Applications need permission to open network sockets

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