UNIVERSITÀ
DEGLI STUDI
DE L'AQUILA

disim

Mobile Programming Laboratory

ANDROID
Http Requests

Teachers

Ing. Tarquini Francesco, Ph.D

Ph.D in Computer Science Engineering

francesco.tarquini@univaq.it

Ing. D’Errico Leonardo

Ph.D Student in Computer Science Engineering

leonardo.derrico@graduate.univaq.it

Teaching Materials

Available on MOODLE platform

http://www.didattica.univaq.it

Google Drive Repository

https://drive.google.com/drive/folders/1ISqZfn0i9Ub3eWNXbvW00rd0hD9ya8OL?usp=sharing

Topics

- Http Requests
  - GET & POST
  - Volley
- JSON Parser

Http Request

Android support Java Class **URLConnection** to do a Http request.

The app needs to access to the Internet network, thus the developer must define the right **<uses-permission>**

How do you create a request? Show the following steps:

- Obtain a new HttpURLConnection;
- Prepare the request;
- Optionally upload a request body;
- Read the response;
- Disconnect.

Http Request

Obtain a new HttpURLConnection

```
URL url = new URL(address);
HttpURLConnection connection = (HttpURLConnection) url.openConnection();
try{
  // Implementation
} finally {
  connection.disconnect();
}
```

Prepare the request

GET

```
connection.setRequestMethod("GET");
connection.setConnectTimeout(THIRTY_SECONDS);
connection.setReadTimeout(THIRTY_SECONDS);
```

POST

```
connection.setRequestMethod("POST");
connection.setConnectTimeout(THIRTY_SECONDS);
connection.setReadTimeout(THIRTY_SECONDS);
connection.setDoOutput(true);
connection.setDoInput(true);
```

Http Request

Optionally upload a request body

```
OutputStream out = connection.getOutputStream();
BufferedOutputStream bos = new BufferedOutputStream(out);
bos.write(body.getBytes());
bos.flush();
bos.close();
```

Read the response

```
InputStream in;
int responseCode = connection.getResponseCode();
if(responseCode == HttpURLConnection.HTTP_OK) {
   in = connection.getInputStream();
} else {
   in = connection.getErrorStream();
}

StringBuilder stringBuilder = new StringBuilder();
BufferedInputStream bis = new BufferedInputStream(in);
byte[] buffer = new byte[1024];
int length = 0;
while((length = bis.read(buffer)) > 0){
   String stringBuffer = new String(buffer, 0 , length);
   stringBuilder.append(stringBuffer);
}
return stringBuilder.toString();
```

Volley

Volley is a HTTP library that makes networking for Android apps easier and faster.

Volley integrates easily with any protocol and comes out of the box with support for raw strings, images, and JSON.

The way to add Volley to your project is to add the following dependency to your app's build.gradle file:

```
implementation 'com.android.volley:volley:1.1.1'
```

Be Careful! It is required the Manifest's Internet permission

Volley - Simple Request

To implement a simple request you must create a Request, obtain the RequestQueue, and add your request at the queue.

- The request works in a separate thread

Volley gives us two different listener to handle the server response and the error.

- Both the results, success or error, work in main thread

```
RequestQueue queue = Volley.newRequestQueue(context);
String url = "your_address";

StringRequest stringRequest = new StringRequest(
        StringRequest.Method.GET,
        url,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
            }
        }, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
    }
});

queue.add(stringRequest);
```

Volley - JSON Request

Volley offers JSON requests

JSONObject

```
String url = "your_address";
JsonObjectRequest request = new JsonObjectRequest(
      JsonObjectRequest.Method.GET,
      url,
      null,
      new Response.Listener<JSONObject>() {
         @Override
         public void onResponse(JSONObject response) {}
      },
      new Response.ErrorListener() {
         @Override
         public void onErrorResponse(VolleyError error) {}
      }
);
queue.add(request);
```

JSONArray

```
String url = "http://www.google.com";
JsonArrayRequest request = new JsonArrayRequest(
      JsonArrayRequest.Method.GET,
      url,
      null,
      new Response.Listener<JSONArray>() {
         @Override
         public void onResponse(JSONArray response) {}
      },
      new Response.ErrorListener() {
         @Override
         public void onErrorResponse(VolleyError error) {}
      }
);
queue.add(request);
```

Volley - Configuration

In the common use case the queue is not the default queue, thus you have to set up a new instance of it defining:

- a cache to handle caching
- a network to perform transport of the requests

```
DiskBasedCache cache = new DiskBasedCache(context.getCacheDir(), CACHE_SIZE);
Network network = new BasicNetwork(new HurlStack());
queue = new RequestQueue(cache, network);
```

It is a best practice of using Volley that you create a RequestQueue as a **singleton**.

To set up the cache for the images you can use ImageLoader

```
ImageLoader imageLoader = new ImageLoader(queue, new ImageLoader.ImageCache() {

    private final LruCache<String, Bitmap> cache = new LruCache<String, Bitmap>(20);

    @Override
    public Bitmap getBitmap(String url) { return cache.get(url); }

    @Override
    public void putBitmap(String url, Bitmap bitmap) { cache.put(url, bitmap); }
});
```

Permissions

The app can allow to the internet network if the Internet Permission is declared in its manifest.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          ...>

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


</manifest>
```

Since the Android Pie v9.0, the security of the users is enhanced, allowed to default only the encrypted connections (as HTTPS). However it is possible to use the cleartext traffic, defining a network security policy:

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
  <domain-config cleartextTrafficPermitted="true">
     <domain includeSubdomains="true">your site</domain>
  </domain-config>
</network-security-config>
```

And declaring in android manifest:

```
android:networkSecurityConfig="@xml/security_policy"
```

JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format.

- Easy for humans to read and write
- Easy for machines to parse and generate

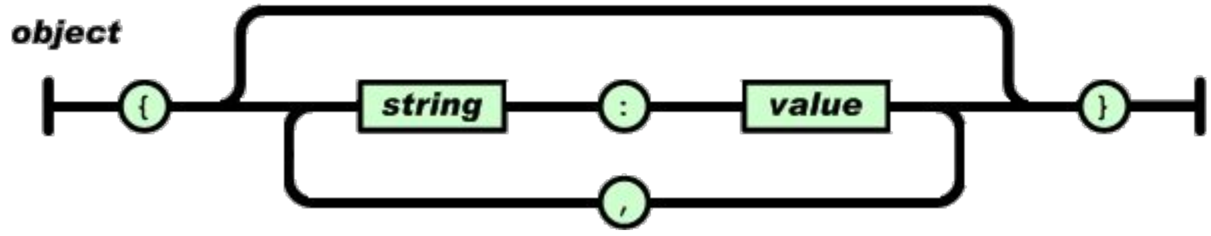
JSON is a text format and it is completely language independent.

JSON is build on two structures:

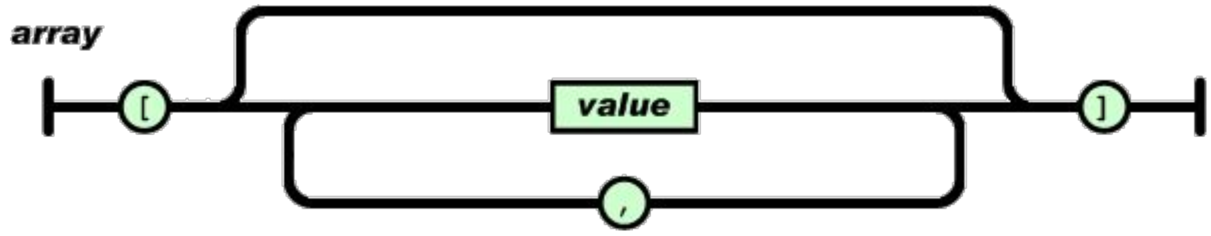
- A collection of name/values pairs, called object
- An ordered list of values, called array

JSON

Object



Array



The value can be a string in double quotes, or a number, or true or false or null, or an object or an array.

JSON

Android includes JSONObject and JSONArray to parse or generate JSON code.

```
{
  "error": false,”students":[
  {“firstname":"Marior","lastname:"Rossim",code:1234},
  {“firstname”:”Pippo","lastname:"Pluto",code:2345}]
}
```

```
JSONObject json = new JSONObject(jsonString);
boolean error = json.getBoolean("error");
if(!error){
   JSONArray array = json.getJSONArray("students");
   for(int i = 0; i < array.length(); i++){
       JSONObject item = array.getJSONObject(i);
       String firstname = item.getString("firstname");
       String lastname = item.getString("lastname");
       long code = item.getLong("code");
   }
}
```

When you use JSON you must handle JSONException.