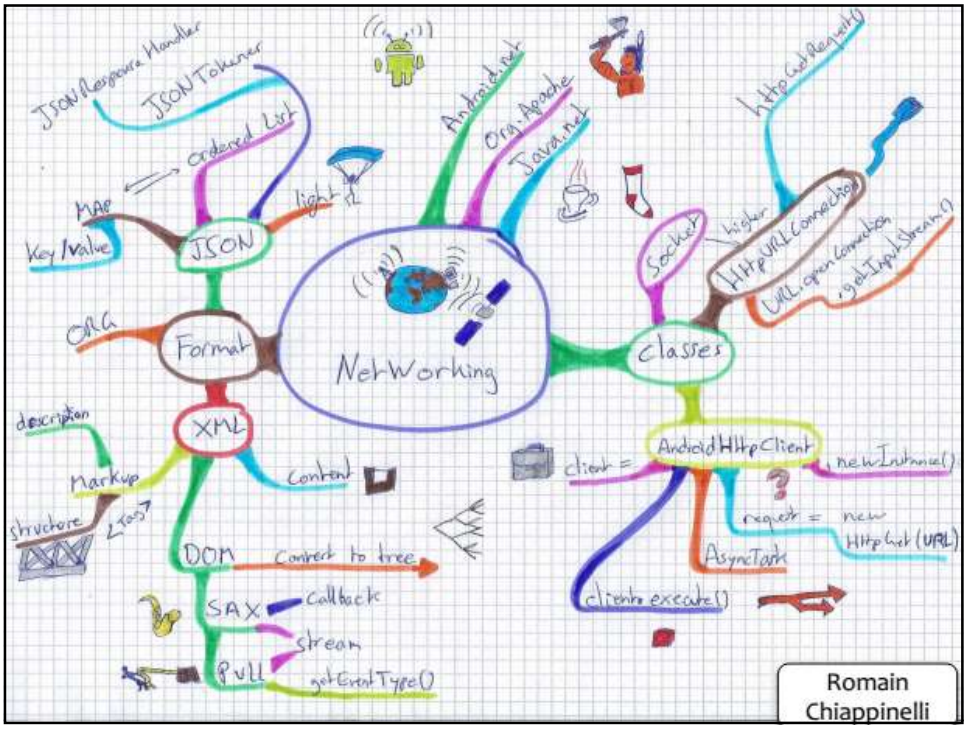
CHAPITRE 9
CONNEXION AU RÉSEAU

• Mr. MEGHAZI

2019-2020

Cours pour les Master II - GL



Contenu

- Connexion au réseau
- Les Classes de connexion au réseau sous Android
- Traitement des réponses HTTP

3

Introduction

- Les premiers appareils portables nous ont permis de la **mobilité** mais ils ont aussi **limité la connectivité**.
- Les appareils d'aujourd'hui fournissent une **meilleure mobilité** et **une meilleure connectivité**.
- Actuellement, ***plusieurs applications*** nécessitent l'utilisation des **données** et des **services** via **Internet**.

4

Connexion au réseau

- Android inclus plusieurs classes de prises en charge de certains mécanismes d'accès au réseau:
  - **java.net** – (Socket, URL)
  - **org.apache** - (HttpRequest, HttpResponse)
  - **android.net** – (URI, AndroidHttpClient, AudioStream)

5

Exemple d'application réseau

- Une application qui envoie une requête à un serveur distant sur des données sismiques, ensuite elle affiche les données demandées.

6

Envoi des requêtes HTTP

- Socket
- HttpURLConnection
- AndroidHttpClient

7

Exemple

8

Exemple – NetworkingSocket (1)

```
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();

            }
        });
    }
```

9

Exemple – NetworkingSocket (2)

```
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";
```

10

Exemple – NetworkingSocket (3)

```
@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;
}
```

11

Exemple – NetworkingSocket (4)

```
@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();
}
```

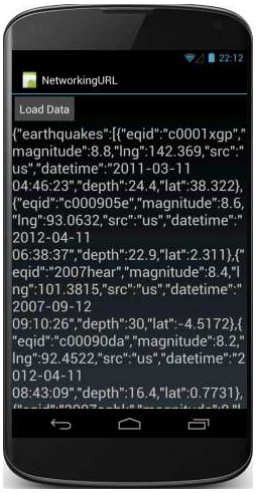
12

HTTPURLConnection

- Permet de travailler dans un niveau plus élevé.
- C'est une API peu flexible que celle du « HttpAndroidClient »

13

Exemple – NetworkingURL (1)



14

Exemple – NetworkingURL (2)

```
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();
            }
        });
    }
```

15

Exemple – NetworkingURL (3)

```
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;
    }
```

16

Exemple – NetworkingURL (4)

```
@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();
}
```

17

AndroidHttpClient

- Une implémentation de la classe Apache «**DefaultHttpClient** ».
- Divise une **transaction** HTTP en deux objets séparés:
  - **Request**
  - **Response**

18

Exemple – NetworkingAndroidHttpClient (1)

```
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();
            }
        });
    }
```

19

Exemple – NetworkingAndroidHttpClient (2)

```
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);

    }
}
```

20

Traitement des réponses Http

- Plusieurs formats populaires, y compris:
  - JSON
  - XML

21

JavaScript Object Notation (JSON)

- Destiné à être un format d'échange de données léger.
- Données encapsulées dans deux types de structures:
  - **Maps** de pairs - Clé/Valeur
  - **Listes** ordonnées de valeurs

http://www.json.org/

22

Données sismiques (JSON Output) - 1

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

23

Données sismiques (JSON Output) - 2

{"earthquakes":

}

24

Données sismiques (JSON Output) - 3

25

Exemple (1)

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));
    }
}
```

26

Exemple (2)

NetworkingAndroidHttpClientJSON

```
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;
    }
}
```

27

eXtensible Markup Language (XML)

- Les documents XML peuvent contenir des **balises** et du **contenu**.
- Les **balises** peuvent encoder une description de la manière de stockage du document et sa structure logique.
- Le **contenu**, c'est tout ce qui reste.

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

28

Données sismiques (XML) - 1

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

29

Données sismiques (XML) - 2

```
<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>
```

30

Analyser du XML

Plusieurs types de parseurs sont disponibles:

- **DOM**: Convertit le document en un arbre.
- **Sax**: Flux pour les appels de l'application.
- **Pull**: Permet à l'application de faire plusieurs itérations sur les entrées XML

31

Exemple (1)

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;
    }
```

32

Exemple (2)

NetworkingAndroidHttpClientXML

```
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;
    }
}
```

33