Florida UNIVERSITÀRIA

# Connexió a la Xarxa en Android

Programació Multimèdia i Dispositius Mòbils

2n. DAM

# Què anem a vore?

- Connectar-nos a la xarxa
- Parsejar un XML
- Parsejar un Objecte JSON

Guia oficial conectivitat:
https://developer.android.com/training/monitoring-device-state/connectivity-monitoring.html

# Connexió a la Xarxa

- Què és?

Ens permet connectar-nos a una URL**.**

http://developer.android.com/training/basics/network-ops/connecting.html

Necessitarem habilitar els següents permisos a l'AndroidManifest.

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

La majoria de les aplicacions d'Android connectades a la xarxa utilitzen HTTP per enviar i rebre dades. Android inclou dos clients HTTP:

- HttpURLConnection i
- Apache HttpClient (deprecat a partir de la versió 6.0 d'Android).

Tots dos suporten el protocol HTTPS, arxius de streaming i descàrregues, temps d'espera configurables, IPv6, i agrupació de connexions.

Es recomana usar HttpURLConnection

# Connexió a la Xarxa

**PREPARACIÓ**

Abans d'intentar connectar-se a la xarxa, s'ha de comprovar si tenim una connexió de xarxa disponible mitjançant getActiveNetworkInfo () i isConnected ().

Penseu, el dispositiu pot estar fora de l'abast d'una xarxa, o l'usuari pot haver deshabilitat tant el Wi-Fi com l'accés de dades mòbils.

```
public void connectatALaXarxa(View view) {
   ...
   ConnectivityManager connMgr = (ConnectivityManager)  getSystemService(Context.CONNECTIVITY_SERVICE);
   NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
   if (networkInfo != null  &&  networkInfo.isConnected()) {
      // podem fer la connexió a alguna URL
   } else {
      // mostrem un error indicant que no disposem de connexió a la Xarxa
   }
   ...
}
```

Es recomana que les connexions a les URL les feu en un **Thread** a part, per a no bloquejar la UI.

# Connexió a la Xarxa

PASSOS A SEGUIR:

http://developer.android.com/reference/java/net/HttpURLConnection.html

1. A partiu d'una URL donada en forma d'un String, construirem un objecte URL.
2. L'objecte URL s'utilitza per establir una connexió mitjançant l'objecte HttpURLConnection. Configurem els paràmetres de la connexió al nostre gust.
3. Una vegada que la connexió s'ha establert, l'objecte HttpURLConnection obté el contingut de la pàgina web mitjançant un InputStream.
4. L'InputStream ha de convertir el fuxe de bytes en algun tipus de dada, el més habitual és convertir-lo a una cadena (si estem rebent codi HTML), o a una Bitmap (si estem rebent una imatge), ...

```
InputStream  is = null;
    try {
        URL url = new URL("https://www.google.es/images/srpr/logo11w.png");
        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        conn.setReadTimeout(10000 /* milisegons */);
        conn.setConnectTimeout(15000 /* milisegons */);
        conn.setRequestMethod("GET");
        conn.setDoInput(true); /* anem a rebre dades */
        // Comença la connexió
        conn.connect();
        int response = conn.getResponseCode();
        InputStream  is = conn.getInputStream();

        // Convertim L'InputStream amb les dades rebudes a un Bitmap
        Bitmap bitmap = BitmapFactory.decodeStream(is);
        ImageView imageView = (ImageView) findViewById(R.id.image_view);
        imageView.setImageBitmap(bitmap);

    // Ens asegurem que l'InputStream es tanca després d'haver-lo utilitzat.
    } finally {
        if (is != null) {
            is.close();
        }
    }
```

```
//llegim de l'InputStream i ho convertim a un String
StringBuilder stringBuilder = new StringBuilder();
BufferedReader reader = new BufferedReader(
                        new InputStreamReader(in));
String line;
while ((line = reader.readLine()) != null) {
    stringBuilder.append(line);
}
```

# PARSEJAR UN XML

<Activitats xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<Pk_Activitat>61668</Pk_Activitat>
<Fk_TemaActivitat>T2</Fk_TemaActivitat>
<Fk_SubTemaActivitat>S0</Fk_SubTemaActivitat>
<Fk_TipusActe>56</Fk_TipusActe>
<Calendari xsi:nil="true"/>
<Internet>1</Internet>
<Pastorets xsi:nil="true"/>
<Pessebres xsi:nil="true"/>
<Activitat_Familiar>0</Activitat_Familiar>
<Data_Inici>2015-02-01T00:00:00</Data_Inici>
<Data_Fi>2015-03-01T00:00:00</Data_Fi>
<Data_Fi_Aproximada>0</Data_Fi_Aproximada>
<Horari/>
<Mes_Informacio/>
<TR3SC>0</TR3SC>
<Entrades/>
<Contacte/>
<Observacions/>
<Descripcio_Imatge/>
<Ruta_Imatge>1Kursaal-Manresa.jpg</Ruta_Imatge>
<Url_Imatge>
http://cultura.gencat.cat/agenda/media/1Kursaal-Manresa.jpg
</Url_Imatge>
<Descripcio_Video_embed/>
<Ruta_Video_Embed/>
<Descripcio_Video_embed_url/>
<Url_video_Embed/>
<Nom_municipi xsi:nil="true"/>
<Nom_comarca xsi:nil="true"/>
<Fk_lloc1>201240</Fk_lloc1>
<Lloc1>Kursaal, Espai d'Arts Escèniques</Lloc1>
<mes_llocs>0</mes_llocs>
<poblacio_localitat>Manresa</poblacio_localitat>
<mes_poblacions>0</mes_poblacions>
<dates_descripcio>De l'01/02/2015 a l'01/03/2015</dates_descripcio>
<data_alta_inet>2015-01-22T06:30:02.370</data_alta_inet>
<denominacio_text>
Programació de febrer al Kursaal, Espai d'Arts Escèniques
</denominacio_text>
<marc_text>Kursaal, Espai d'Arts Escèniques (Manresa) Febrer</marc_text>
<descripcio_text>
Espectacle Bombollavà A càrrec de Pep Bou Diumenge 1 de febrer, 18 h Preu: 18 € Actuació de Merche Música. Presenta el seu nou disc Quiero contarte Divendres 6 de febrer, 21 h Preu: 22 € Companyia 2princesesbarbudes, amb Enciclopèdia baixeta de la nit Espectacle infantil Dissabte 7 de febrer, 17.30 h Preu: 8 € Representació de La ratonera, d'Agatha Christie Teatre. A càrrec de Mariona Ribas, Ferran Carvajal, Aleix Rengel, Xavier Bertran, Anna Gras-Carreño, Santi Ibáñez, Joan Amargós i Isabel Rocatti Dissabte 7 de febrer, 18 i 21 h Diumenge 8 de febrer, 18 h .....

# Parsejar Dades XML

- Hi han diferents analitzadors(Parsers) per poder interpretar els XML.
- Java ens ofereix 2:
  - DOM - Construeix l'arbre de nodes en memòria
  - SAX - A mesura que rep l'XML el va analitzant
- Android ens peremet utilitzar els parsers de Java i a més a més ens recomana utilitzar el seu pròpi
  - XmlPullParser – Versió més eficient que SAX en dispositius mòbils.

# Parsejar Dades XML

- PASSOS:

```
try {
        //Instanciem un objecte XmlPullParser, i comencem a analitzar l'xml
        XmlPullParser parser = Xml.newPullParser();
        //Indiquem que no volem processar els namespaces
        parser.setFeature(XmlPullParser.FEATURE_PROCESS_NAMESPACES, false);
        //Indiquem que l'entrada de l'XML la rebrem des d'un InputStream (in)
        parser.setInput(in, null);
        //Iniciem el procés de parseig
        parser.nextTag();
        //Tornem la informació obtinguda del parseig. El mètode extrauInfodeXml és
        // qui obtindrà la informació dels tags que ens interesse, i la guardarà en una
        //Estructura de dades adient.
        return extrauInfodeXml(parser);

    } finally {
        in.close();  //ens asegurem de tancar el InputStream
    }
```

# Parsejar Dades XML

- PROCESSEM LES TAGS DESSITJADES

```
private List readFeed(XmlPullParser parser) throws XmlPullParserException, IOException {
   List entries = new ArrayList();  //On guardarem les dades a tornar

   parser.require(XmlPullParser.START_TAG, ns, "AGENDA_CULTURAL"); //INICI DOCUMENT
   while (parser.next() != XmlPullParser.END_TAG) {
      if (parser.getEventType() != XmlPullParser.START_TAG) {
         continue;
      }
      String name = parser.getName();
      // Busquem l'etiqueta (TAG) "Activitats"
      if (name.equals("Activitats")) {
         entries.add(llegirActivitat(parser));  // llegirActivitat: Mètode que ens treu les dades que
                                                // volem del tag Activitats
      }
   }
   return entries;
}
```

# Parsejar Dades XML

- Dins del TAG <Activitats>, volem extreure les dades de <Lloc1>, <dates_descripcio>, <descripcio_text>
- Crearem els següents mètodes:
  - Un mètode per a llegir cada TAG dels que colem extreure dades: readLloc1(), readDatesDescripcio() … .
  - El parser anirà llegint TAGs des de l'InputStream, I quan trobe una que ens interesse, cridarà al mètode corresponent.
  - Extraurem les dades de cada tag cridant a mètodes distints, depenent de si les dades es troben com a atributs del tag, o com a text del tag:
    - parser.getText().
    - parser.getAttributeValue()

# Parsejar Dades XML

- 
```
private Activitat llegirActivitat(XmlPullParser parser) throws XmlPullParserException, IOException {
    parser.require(XmlPullParser.START_TAG, ns, “Activitat");
    String lloc= null;
    String data= null;
    String descripcio= null;
    while (parser.next() != XmlPullParser.END_TAG) {
        if (parser.getEventType() != XmlPullParser.START_TAG) {
            continue;
        }
        String name = parser.getName();
        if (name.equals(“Lloc1")) {
            title = readLloc1(parser);
        } else if (name.equals(" dates_descripcio ")) {
            summary = readDadesDescripcio(parser);
        } else if (name.equals(" descripcio_text ")) {
            link = readDescripcioText(parser);
        }
    }
    return new Activitat(lloc, data, descripcio);
}

// Processes Lloc1 tags i
private String readLloc1(XmlPullParser parser) throws IOException, XmlPullParserException {
    parser.require(XmlPullParser.START_TAG, ns, “Lloc1");
    String lloc= readText(parser);
    parser.require(XmlPullParser.END_TAG, ns, “Lloc1");
    return lloc;
}
```

```
public static class Activitat{
    public final String lloc;
    public final String data;
    public final String descripcio;

    private Entry(String ll, String da, String des) {
        this.lloc = ll;
        this.data = da;
        this.descripcio = des;
    }
}
```

# PARSEJAR UN JSON

# JSON

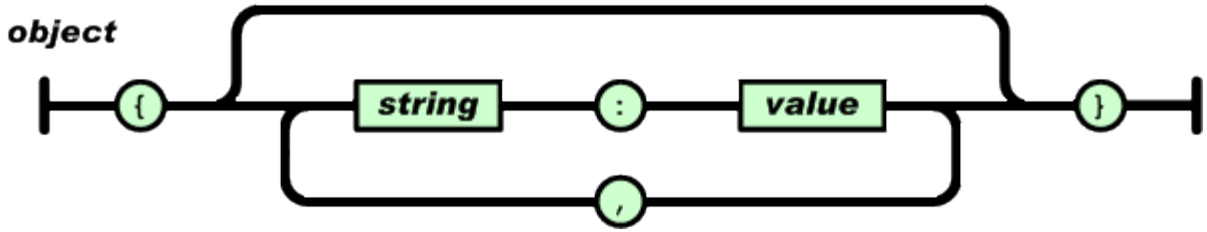

- **Què és JSON?** (acrònim de **JavaScript Object Notation**) és un estàndard obert basat en text dissenyat per a intercanvi de dades llegible per humans.
- Deriva del llenguatge JavaScript, per a representar estructures de dades simples i llistes associatives, anomenades objectes. Malgrat la seua relació amb el JavaScript, té implementacions per a gran part dels llenguatges de programació.
- S'utilitza principalment per intercanviar dades entre un servidor i una aplicació web, sent una alternativa a l'XML. S'utilitza freqüentment en aplicacions Ajax.

# JSON


- **Per què JSON?**
- L'eficiència en l'ús dels recursos a l'hora de desenvolupar aplicacions mòbils és una restricció que no podem obviar.
- En aquest cas intentarem reduir el consum de dades d'internet (optimitzat el màxim possible) per estalviar bateria i aprofitar les tarifes de dades existents actualment.
- En aquest sentit JSON, gràcies a la seua estructura, estalvia considerablement aquestos recursos front a XML.

# JSON


- Format JSON

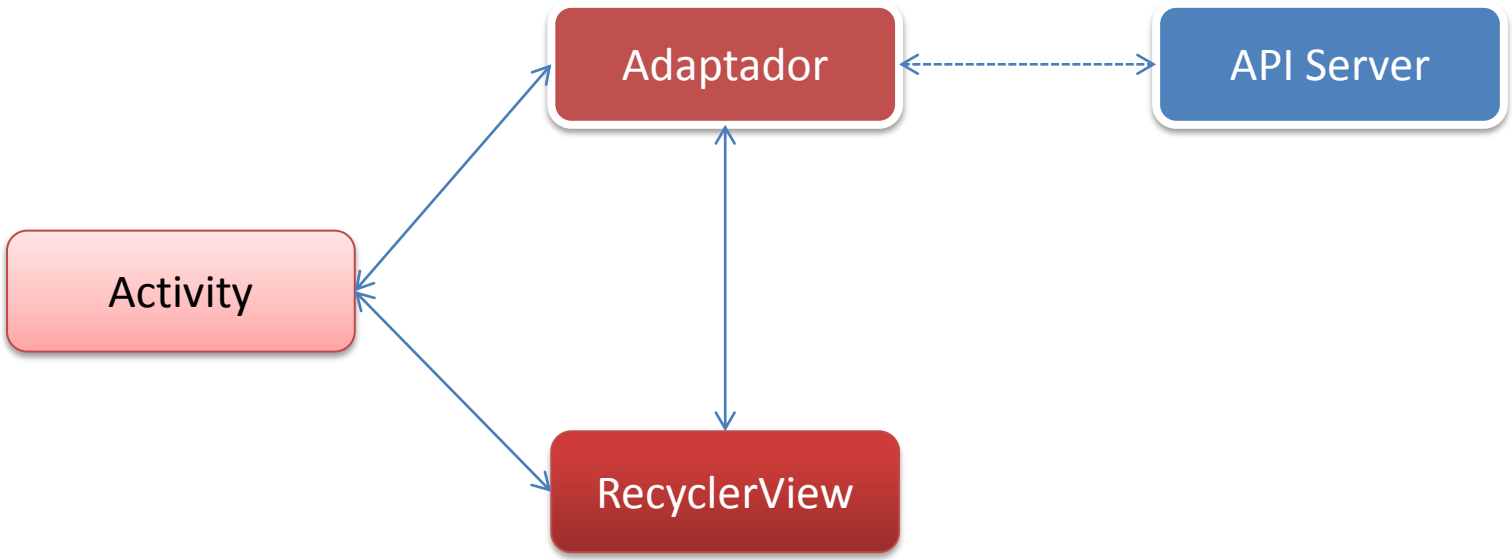



- Referència:

  http://www.json.org/json-es.html

```
{ "empinfo" :
   {
        "employees" : [
        {
          "name" : "Scott Philip",
          "salary" : £44k,
          "age" : 27,

        },

        {
          "name" : "Tim Henn",
          "salary" : £40k,
          "age" : 27,
        },

        {
         "name" : "Long Yong",
         "salary" : £40k,
         "age" : 28,

        }
                  ]
   }
}
```

# Arquitectura típica




- L'activity conté un RecyclerView i un Adaptador.
- L'adaptador obté les dades a través d'una connexió de xarxa, i enviant una/es pecitió/ons a un API Server
- L'adaptador manipula eixes dades per a adaptar-les al format del ListView

# Problemes

- Totes les connexions es fan en serie (una rere l'altra).
- Si rotem el terminal, l'activity es destrueix, i tornem a començar


```
- response: {
  - groups: [
    - {
        type: "places",
        name: "Matching Places",
      - items: [
        - {
            id: "4be40357d27a20a1aa75935b",
            name: "Indira Gandhi International Airport (DEL)",
            contact: { },
          - location: {
              address: "Indira Gandhi International Airport",
              city: "New Delhi",
              state: "India",
              lat: 28.55448289261816,
              lng: 77.0896053314209,
              distance: 12291
            },
          - categories: [
            - {
                id: "4bf58dd8d48988d1ed931735",
                name: "Airport",
                pluralName: "Airports",
                icon: "https://foursquare.com/img/categories/travel/airport.png",
              - parents: [
                    "Travel Spots"
                ],
                primary: true
              },
            - {
                id: "4bf58dd8d48988d1eb931735",
                name: "Airport Terminal",
                pluralName: "Airport Terminals",
                icon: "https://foursquare.com/img/categories/travel/airport_terminal.png",
              - parents: [
                    "Travel Spots",
                    "Airports"
                ]
              }
            ],
            verified: true,
          - stats: {
              checkinsCount: 3724,
              usersCount: 1838
            },
          - hereNow: {
              count: 0
            }
          },
        - {
            id: "4b6f9e1ef964a52010f82ce3",
            name: "Gurgaon Delhi Toll Gate",
```

# JSON Consideracions

- De la mateixa manera que que per a parsejar un XML disposem de dos models (DOM/SAX), per a parsejar documents JSON també tenim dues alternatives:
- Classe JSONObject – Carrega tot l'objecte JSON en memòria (disp. des de  API 1.0)
- Classe **JSONReader** – Proporcionada per Google, parseja les dades conforme li van arribant (com SAX) (disp. des de API 11)

# JSON Consideracions

- Per tant, per a parsejar documents JSON molt grans és recomanable utilitzar JSONReader.

```
{
   "contactes": [
      {
            "id": ”c001",
            ”nom": ”Manel",
            "email": ”manel@gmail.com",
            "addreça": "xx-xx-xxxx,x - street, x - country",
            ”genere" : ”mascle",
            ”telefon": {
               "mobil": ”6000000000",
               ”casa": ”+34 96 000000",
               "oficina": "00 000000"
            }
      },
      .
      .
 ]
}
```

# JSON Consideracions

- En què es diferencia [ i { ?

[ ⟷ representa JSONArray

{ ⟷ representa JSONObject

Per tant, necessitarem cridar al mètode adient per accedir a les dades.

[ ⟷ getJSONArray()

{ ⟷ getJSONObject()

```
{
  "contactes": [
    {
        "id": ”c001",
        ”nom": ”Manel",
        "email": ”manel@gmail.com",
        "addreça": "xx-xx-xxxx,x - street, x - country",
        ”genere" : ”mascle",
        ”telefon": {
          "mobil": ”6000000000",
          ”casa": ”+34 96 000000",
          "oficina": "00 000000"
        }
    },
    .
    .
 ]
}
```

# Parsejar un JSON


EXEMPLE

- Com a exemple, anem a utilitzar un fitxers que ens proporciona jsonplaceholder que és un simulador de dades falses per a JSON

- Ens connectarem al següent servei web (REST) https://jsonplaceholder.typicode.com/users

# Parsejar un JSON


```
[
{},
{},
{},
{},
{},
{},
{},
{},
{},
{}
]
```

```
[ {
"id": 1,
"name": "Leanne Graham",
"username": "Bret",
"email": "Sincere@april.biz",
"address": { "street": "Kulas Light",
             "suite": "Apt. 556",
             "city": "Gwenborough",
             "zipcode": "92998-3874",
             "geo": { "lat": "-37.3159",
                      "lng": "81.1496”
                    }
           },
"phone": "1-770-736-8031 x56442",
"website": "hildegard.org",
"company": { "name": "Romaguera-Crona",
             "catchPhrase": "Multi-layered client-server neural-net",
             "bs": "harness real-time e-markets”
           }
},
{},
```

Florida UNIVERSITÀRIA

# Parsejar un JSON


JSONArray

JSONObject

```
[ {
"id": 1,
"name": "Leanne Graham",
"username": "Bret",
"email": "Sincere@april.biz",
"address": { "street": "Kulas Light",
             "suite": "Apt. 556",
             "city": "Gwenborough",
             "zipcode": "92998-3874",
             "geo": { "lat": "-37.3159",
                      "lng": "81.1496"
                    }
           },
"phone": "1-770-736-8031 x56442",
"website": "hildegard.org",
"company": { "name": "Romaguera-Crona",
             "catchPhrase": "Multi-layered client-server neural-net",
             "bs": "harness real-time e-markets"
           }
},
...
{}
]
```

Florida UNIVERSITÀRIA

# Què cal fer en Android per Parsejar un JSON?

Descarregueu l'exemple:
https://github.com/mvielcor/XarxaJSON.git

Passos:

1. Comprovar que hi tenim connexió a la Xarxa.
2. Connectar-nos al servei web que ens proporcionarà el document JSON.
3. Parsejar el document JSON i extreure la informació que necessitem.
4. Gestionar eixa informació (generalment mostrant-la en un RecyclerView).

A partir de la versió 5.0 (API 21), Android no permet fer connexions a la xarxa al UI Thread, per tant haurem de crear un **tasca asíncrona (AsyncTask)** per fer els passos 2,3 i 4

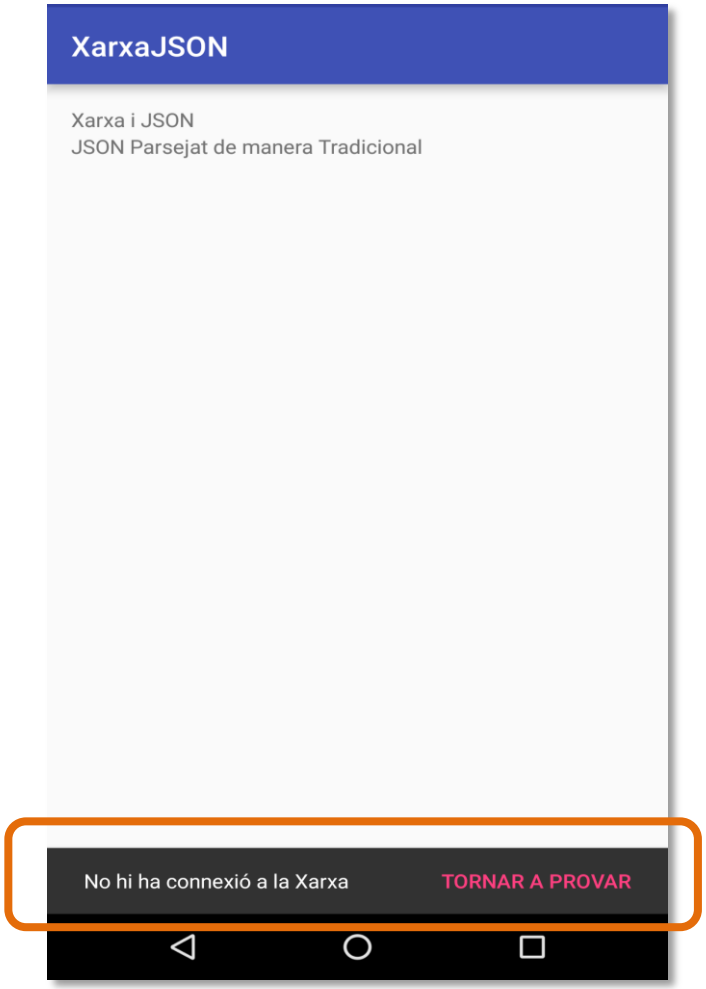
# Pas 1. Comprovar que hi tenim connexió a la Xarxa

```
protected boolean comprovaConnexio(){
   ConnectivityManager connMgr = (ConnectivityManager)  getSystemService(Context.CONNECTIVITY_SERVICE);
   NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
   if (networkInfo != null  &&  networkInfo.isConnected()) {
        //podem fer la connexió a alguna URL
        hiHaXarxa=true;
       // AsyncTask que fa els passos 2,3 i 4
        new ConnectaURL().execute("https://jsonplaceholder.typicode.com/users");
        return true;
   } else {
       // mostrem un error indicant que no disposem de connexió a la Xarxa
        Snackbar mySnackbar = Snackbar.make(findViewById(R.id.relativeLayout),
           "No hi ha connexió a la Xarxa", Snackbar.LENGTH_INDEFINITE);
       // li afegim un botó per tornar a comprovar si hi tenim connnexió
        mySnackbar.setAction("Tornar a provar", new View.OnClickListener() {
        @Override
            public void onClick(View view) {
                hiHaXarxa = comprovaConnexio();
            }
        });
        mySnackbar.show();
        return false;
   }
}
```

Atribut booleà de la classe

SnackBar amb un botó d'acció

# Pas 1. Comprovar que hi tenim connexió a la Xarxa




SnackBar amb un botó d'acció

# Pas 2. Connectar-nos al servei web

```
private String connectaURL(String llocAConnectar){
   URL url;
   String resposta=null;
   try {
      url = new URL(llocAConnectar);
      HttpURLConnection conn = (HttpURLConnection) url.openConnection();
      conn.setRequestMethod("GET"); // Podrà ser PUT, POTS,GET,DELETE,HEAD,OPTIONS,TRACE, vegeu api
      conn.setDoInput(true); /* anem a rebre dades */
       // Comença la connexió
      conn.connect();
      int response = conn.getResponseCode();  //rebem el codi de resposta que envia el servidor
      InputStream  is = new BufferedInputStream(conn.getInputStream());
      resposta = converteixStreamAString(is);  // Hem de convertir l'Stream a un String, per a manipular-lo

   } catch (MalformedURLException e) {
      e.printStackTrace();
   } catch (ProtocolException e) {
      e.printStackTrace();
   } catch (IOException e) {
      e.printStackTrace();
   }finally{
      return resposta;
   }
}
```

# Pas 2. (continuació)

```
private String converteixStreamAString(InputStream is) {
    BufferedReader reader = null;  // buffer on anirem llegint el document JSON que ens envie el servidor
    StringBuilder sb = new StringBuilder(); // Cadena on anirem afegint les dades del document JSON
                                            //conforme les anem llegint del buffer, línia a línia
    try {
      // creem un buffer per a anar llegint del InputStreamReader
      reader = new BufferedReader(new InputStreamReader(is));
      String line;
      // Bucle per a llegir totes les línies que ens envia el servidor
      while ((line = reader.readLine()) != null) {
        sb.append(line).append('\n');  // afegim la línia llegida a la cadena
      }
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
      try {
        is.close();  // Tanquem l'InputStreamReader
        reader.close(); // Tanquem el BufferReader
      } catch (IOException e) {
        e.printStackTrace();
      }
    }
    return sb.toString();
}
```

L'objectiu d'aquest mètode és anar llegint línies de l'Stream d'entrada, i ana construint un String amb la informació rebuda des del servidor.

# Pas 3. Parsejar el document JSON

```
private void parsejaJSONManeraTradicional(String documentJSON) {
   // Rebem el document JSON i parsejem les dades rebudes al nostre gust
   if (documentJSON != null) {
      try {
         // Aquest document d'exemple, comença amb un JSONArray
         JSONArray contacts =  new JSONArray(documentJSON);
         // bucle per a recórrer tots els Contactes
         for (int i = 0; i < contacts.length(); i++) {
              //Ací realitzarem el parsejat (vegeu pàgina següent)
         }
      } catch (final JSONException e) {
            Snackbar.make(findViewById(R.id.relativeLayout),
               "Error parsejant Json", Snackbar.LENGTH_LONG).show();
      }
   } else {
      Snackbar.make(findViewById(R.id.relativeLayout),
            "Error intentant rebre el Json.", Snackbar.LENGTH_LONG).show();
   }
}
```

# Pas 3. (continuació)

```
JSONObject jsonObj = contacts.getJSONObject(i);  //Agafem un contacte
//accedim als camps que ens interessa de l'objecte JSON
String id = jsonObj.getString("id");
String name = jsonObj.getString("name");
String email = jsonObj.getString("email");
String address = jsonObj.getJSONObject("address").getString("street");
// Company és un altre JSONObject
JSONObject company = jsonObj.getJSONObject("company");
String company_name = company.getString("name");
String company_catchPhrase = company.getString("catchPhrase");
// Creem un Contacte temporal per a afegir-lo a l'ArrayList
Contacte unContacte = new Contacte();
// Omplim les dades del Contacte amb les dades obtingudes del JSONObject
unContacte.setId(Integer.parseInt(id));
unContacte.setName(name);
unContacte.setEmail(email);
unContacte.setCompany_name(company_name);
unContacte.setCompanyCatchPhrase(company_catchPhrase);
// AFEGIM EL CONTACTE A L'ARRAYLIST
contactesParsejats.add(unContacte);
```

# Pas 3. (continuació)

jsonObj

JSONArray

company

```
{
"id": 1,
"name": "Leanne Graham",
"username": "Bret",
"email": "Sincere@april.biz",
"address": { "street": "Kulas Light",
             "suite": "Apt. 556",
             "city": "Gwenborough",
             "zipcode": "92998-3874",
             "geo": { "lat": "-37.3159",
                      "lng": "81.1496"
                    }
           },
"phone": "1-770-736-8031 x56442",
"website": "hildegard.org",
"company": { "name": "Romaguera-Crona",
             "catchPhrase": "Multi-layered client-server neural-net",
             "bs": "harness real-time e-markets"
           }
},
```

# Pas 4. Gestionar eixa informació

```
@Override
protected void onPostExecute(String s) {
    super.onPostExecute(s);
    //Mostrem el temps que s'ha tardat en realitzar el parseig
    tv_etiqueta1.append(" realitzat en "+s);
    //Creem l'adaptador que interactuarà amb les dades
    ac = new AdaptadorContacte(contactesParsejats);
    //Enllacem el RecyclerView amb l'adaptador per a que mostre el contingut del Recyclerview
    rvTradicional.setAdapter(ac);
}
```

# PARSEJAR UN JSON amb GSON

- **Què és?**

Gson és una llibreria de còdi obert proporcionada per Google per a ***parsejar*** i ***generar*** documents JSON.

GSON és molt fàcil d'utilitzar, encara que ja que utilitzem llibreries de tercers, cal comentar que hi ha altres opcions (millors?) com **Jackson** o **Boon**

**http://wiki.fasterxml.com/JacksonHome**

**https://github.com/boonproject/boon/wiki/Boon-JSON-in-five-minutes**

- Necessitem:

Afegirem la dependència:

compile 'com.google.code.gson:gson:2.3.1’

al gradle del projecte.

- Si el document JSON a parsejar és el següent:
- { "frutas":
  [
  { "nombre_fruta":"Manzana" , "cantidad":10 },
  { "nombre_fruta":"Pera" , "cantidad":20 },
  { "nombre_fruta":"Naranja" , "cantidad":30 }
  ]
  }

- Prepararem la classe *Fruta* per a treballar posteriorment amb els nostres objectes:

```
public class Fruta {
    public String nombre;
    public int unidades;
    public Fruta (String nombre, int unidades){
            this(nombre, unidades);
    }
 // Mètodes get
...
}
```

- Només ens queda Parsejar la nostra cadena JSON:

```
String json = CADENA_JSON;
Gson gson = new Gson();
List<Fruta> frutas = gson.fromJson(CADENA_JSON, Fruta.class);
```

# Utilitats

- Amb GSON es poden fer moltes més coses. Reviseu:

http://howtodoinjava.com/best-practices/google-gson-tutorial-convert-java-object-to-from-json/

- I si el document JSON no és tan senzill com el de l'exemple?
- És molt costós crear una classe per a cada element d'un JSON gran!!!!
- Per això tenim: http://jsongen.byingtondesign.com/

Doneu algun donatiu si ho fer servir!!!

# Accedir variables PUT/DELETE amb PHP

L'accés a les variables de les peticions HTTP fetes mitjançant el mètode POST, en PHP, el fem mitjançant $_POST.

L'accés a les variables de les peticions HTTP fetes mitjançant el mètode GET en PHP, el fem mitjançant $_GET.

No hi ha cap variable (en PHP per accedir a les peticions PUT ni DELETE)

PHP pot llegir eixes variables mitjançant un stream d'entrada. Podem accedir a eixe stream mitjançant **php://input.**

Així, la funció **file_get_contents("php://input");** ens torna un string amb les variables passades. Ara només ens caldria parsejar eixes variables i guardar-les en algun lloc per accedir a cadascuna d'elles quan ens interese.

**parse_str()** ens fa eixa feina!!!

**parse_str(**file_get_contents("php://input")**,**$dadesRebudes**);** Ens llig les variables rebudes mitjançant l'stream d'entrada, i parse_str() ens les parseja i les guarda en un array anomenat $dadesRebudes.

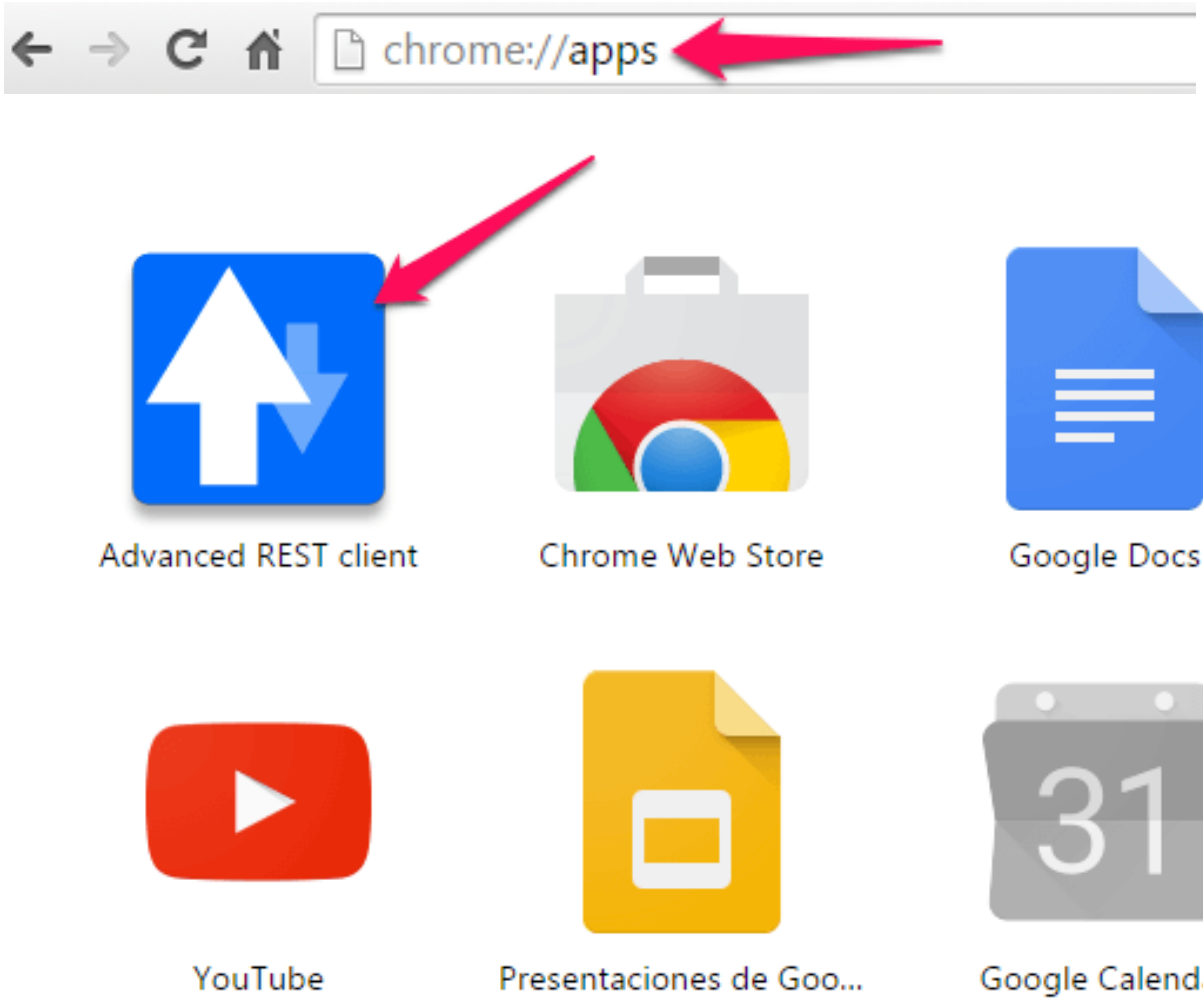
A partir d'ara podem accedir a qualsevol variable posant **$dadesRebudes[**'nom_de_la_variable'**]**

# Accedir variables PUT/DELETE amb PHP

Exemple:

```
if($_SERVER['REQUEST_METHOD'] == 'GET') {
    echo “petició GET\n";
    echo $_GET['fruita']." és una fruita\n";
    echo “Vull".$_GET['quantitat']." kg.\n\n";
} elseif($_SERVER['REQUEST_METHOD'] == 'PUT') {
    echo “petició PUT\n";
    parse_str(file_get_contents("php://input"),$dades);
    echo $dades['fruita']." és una fruita\n";
    echo “Vull".$dades['quantitat']." kg.\n\n";
}
```

# Client REST per a chrome



Advanced Rest Client

[Unnamed] Save Open

http://localhost/api.peopleapp.com/v1/usuarios/registro ← url

GET POST PUT PATCH DELETE HEAD OPTIONS Other

Raw Form **Headers**

método

Request
Socket
Projects
Saved
History
Settings
About

Rate this application ♥
Donate

texto plano

Raw Form Files (0) **Payload**

Encode payload Decode payload

```
{
   "nombre":"carlosl",
   "contrasena":"12345",
   "correo":"carlos@mail.com"
}
```

cuerpo

enviar petición

application/json *Set "Content-Type" header to overwrite this value.*

header Content-Type

Clear Send