JSON Parser

# JSON

JSON (JavaScript Object Notation) là một chuẩn thuần văn bản và có cấu trúc để giúp việc truyền tải dữ liệu.

Nó có nguồn gốc từ ngôn ngữ kịch bản javascript để trình bày cấu trúc dữ liệu đơn giản kết hợp với mảng, được gọi là object. Dù nó có mối quan hệ với javascript nhưng nó là ngôn ngữ độc lập có thể được phân tích trong các ngôn ngữ khác.

JSON có các kiểu dữ liệu: Number, String, Boolean, Array, Object, null.

Mục đích của JSON cũng giống như XML tuy nhiên cách viết ngắn hơn và cách phân tích (parser) cũng dễ hơn.

# Cú pháp JSON

Cú pháp json theo các qui định sau:

* Dữ liệu gồm các cặp tên / giá trị
* Dữ liệu được phân cách dấu phẩy “,”
* Dấu ngoặc nhọn “{ }” dùng để giữ các đối tượng (object).
* Dấu ngoặc vuông “[ ]” dùng để giữ các mảng (array)

## Cặp tên / giá trị

Dữ liệu được viết là các cặp tên / giá trị. Một cặp tên / giá trị gồm 1 trường tên đặt trong dấu nháy kép, tiếp theo là dấu hai chấm và tiếp theo là giá trị (giá trị có dấu nháy kép hay không tùy theo kiểu dữ liệu).

“ten”: “teo”

Diễn giải ra nghĩa là trường tên có giá trị là teo

“tuoi”:20

Diễn giải ra nghĩa là trường tuổi có giá trị là 20. Số 20 không có nhấy kép do 20 là dữ liệu kiểu số.

## Đối tượng JSON (JSON Object)

Đối tượng json được viết trong cặp dấu ngoặc nhọn “ { } “. Trong một đối tượng có có nhiều cặp tên / giá trị phân cách nhau bằng dấu phẩy “,”.

{ “ten” : ”teo” , “tuoi” : 20 }

## Mảng JSON (JSON Array)

Mảng json được bọc trong dấu ngoặc vuông “ [ ] “. Một mảng có thể chứa nhiều đối tượng (object).

{

“nhanvien” : [

{ “ten” : “teo” , “tuoi” : 20 },

{ “ten” : “ti” , “tuoi” : 25 },

{ “ten” : “tin” , “tuoi” : 35}

]

}

So với cách viết của XML (bên dưới), Json viết ngắn hơn.

<root>

<nhanvien>

<ten>teo</ten>

<tuoi>20</tuoi>

</nhanvien>

<nhanvien>

<ten>ti</ten>

<tuoi>25</tuoi>

</nhanvien>

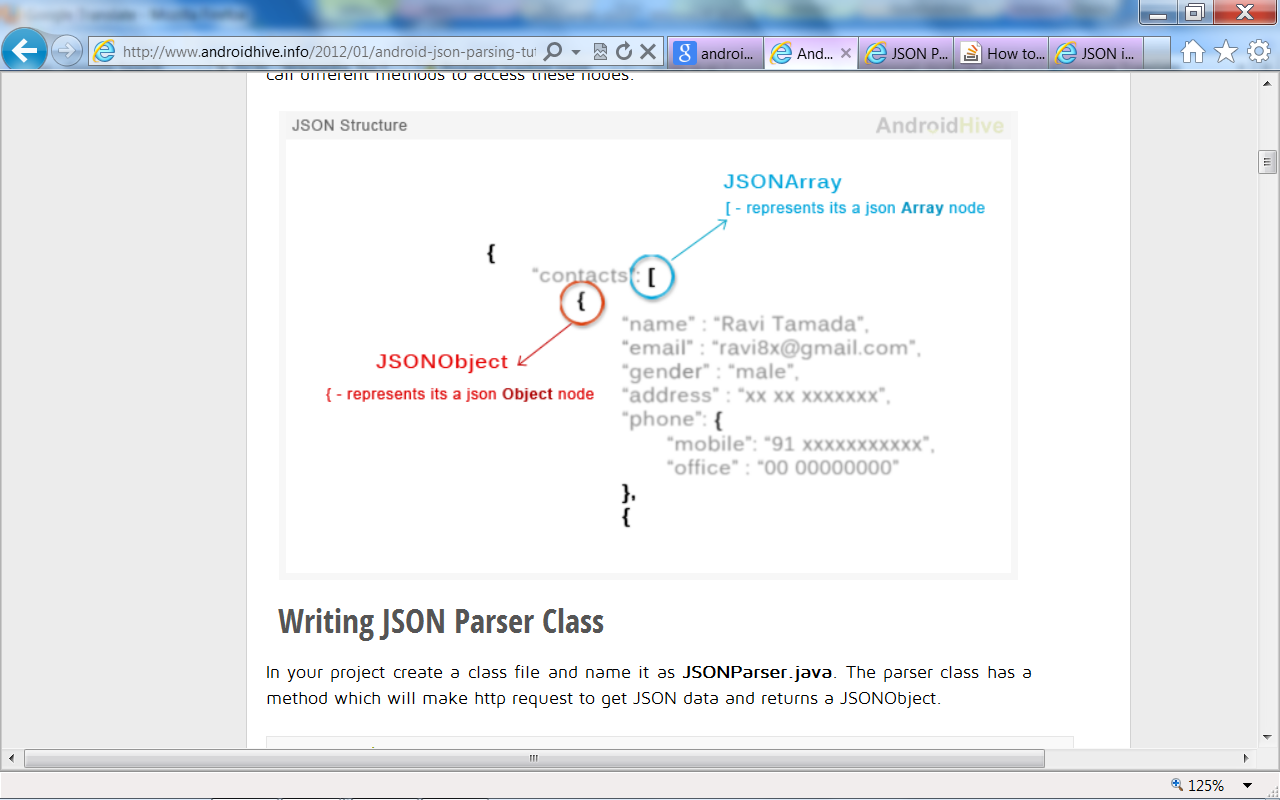
<nhanvien>

<ten>tin</ten>

<tuoi>35</tuoi>

</nhanvien>

</root>



# JSON và Android

JSON cũng giống như là XML nên cũng có thể dùng thao tác làm việc với HTTP để đọc dữ liệu từ server xuống. Sau khi đọc xong ta sẽ phân tích JSON như sau:

### Tạo đối tượng JSONObject

JSONObject jsonobject =new JSONObject(result);

### Lấy chuỗi

String jsonstring= jsonobject.getString(“TENCHUOI”);

### Lấy boolean

String jsonboolean=jsonobject.getBoolean(“TENBOOLEAN”);

### Lấy mảng

JSONArray jsonarray=jsonobject.getJSONArray(“ARRAYNAME”);

### Lấy Item từ mảng

for(int i=0 ; i<jsonarray.length( ) ; i++)

{

JSONObject oneobject=jsonarray.getJSONObject(i);

String item1=oneobject.getString(“STRINGNAMEINARRAY”);

String item1=oneobject.getString(“STRINGNAMEINARRAY”);

}

Thực hành

# Bài 1: Viết chương trình đọc 1 file JSON từ server và phân tích dữ liệu để hiển thị.

1. Chép file “contacts” lên localhost.
2. Tạo project (version 1.6), đặt tên “docfilejson1”.
3. Mở file “manifest” thêm dòng phân quyền như sau:

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

1. Để thuận tiện ta sẽ tạo ra 1 class mới phục vụ cho việc đọc json. Tạo class mới đặt tên vd: “JSONParser.java”. Trong class vừa tạo khai báo cá biến toàn cục:

InputStream is=**null**;

JSONObject jobj=**null**;

String json;

1. Tiếp theo ta xây dựng hàm tên getJSONFromUrl. Hàm này nhận vào một chuỗi là đường dẫn dẫn đến file json trên server. Nó sẽ trả về chuỗi json.

**public** JSONObject getJSONFromUrl(String url)

{

**try**{

//khoi tao

DefaultHttpClient client=**new** DefaultHttpClient();

HttpPost httppost=**new** HttpPost(url);

//thuc thi, lay ve noi dung

HttpResponse httpresponse=client.execute(httppost);

HttpEntity httpentity=httpresponse.getEntity();

is=httpentity.getContent();

//doc du lieu

BufferedReader reader=

**new** BufferedReader(**new** InputStreamReader(is,"iso-8859-1"),8);

StringBuilder sb=**new** StringBuilder();

String line=**null**;

**while**((line=reader.readLine())!=**null**)

{

sb.append(line+"\n");

}

is.close();

json=sb.toString(); //doc StringBuilder vao chuoi

jobj=**new** JSONObject(json); //dua chuoi vao doi tuong json

}**catch**(Exception e)

{

Log.*d*("loi", e.toString());

}

**return** jobj;

}

1. Trở về file java chính. Khởi tạo 1 số biến toàn cục trong class như sau:

**private** **static** String *url* = "http://10.0.2.2:8080/testandroid/contacts";

// JSON Node names

**private** **static** **final** String *TAG\_CONTACTS* = "contacts";

**private** **static** **final** String *TAG\_ID* = "id";

**private** **static** **final** String *TAG\_NAME* = "name";

**private** **static** **final** String *TAG\_EMAIL* = "email";

**private** **static** **final** String *TAG\_ADDRESS* = "address";

**private** **static** **final** String *TAG\_GENDER* = "gender";

**private** **static** **final** String *TAG\_PHONE* = "phone";

**private** **static** **final** String *TAG\_PHONE\_MOBILE* = "mobile";

**private** **static** **final** String *TAG\_PHONE\_HOME* = "home";

**private** **static** **final** String *TAG\_PHONE\_OFFICE* = "office";

// contacts JSONArray

JSONArray contacts = **null**;

1. Trong hàm onCreate ta khởi tạo đối tượng JSONParser (đã viết class ở trên) sau đó dùng hàm getJSONFromUrl để lấy về chuỗi json. Sau đó ta tạo mảng chứa các contact. Dùng for để duyệt tất cả các phần tử contact. Tương ứng mỗi phần tử trong mảng contact ta tạo ra đối tượng JSONObject rồi dùng hàm getString để lấy dữ liệu ra. Lưu ý: Đối với “phone” nó lại là một object chứa các phần tử con là “mobile”, “home”, “office” nên ta phải lấy ra đối tượng JSONObject “phone” sau đó mới lấy tiếp dữ liệu con từ nó.

JSONParser jsonparse=**new** JSONParser();

//lay doi tuong json

JSONObject jsonobject=jsonparse.getJSONFromUrl(*url*);

**try**{

//tao mang contacts

contacts=jsonobject.getJSONArray(*TAG\_CONTACTS*);

//duyet tung contacts

**for**(**int** i=0;i<contacts.length();i++)

{

//lay ra mot doi tuong contact

JSONObject onecontact=contacts.getJSONObject(i);

//lay du lieu

String id=onecontact.getString(*TAG\_ID*);

String name=onecontact.getString(*TAG\_NAME*);

String email=onecontact.getString(*TAG\_EMAIL*);

String address=onecontact.getString(*TAG\_ADDRESS*);

String gender=onecontact.getString(*TAG\_GENDER*);

//so phone la la json Object

JSONObject phone=onecontact.getJSONObject(*TAG\_PHONE*);

String mobile=phone.getString(*TAG\_PHONE\_MOBILE*);

String home=phone.getString(*TAG\_PHONE\_HOME*);

String office=phone.getString(*TAG\_PHONE\_OFFICE*);

Log.*d*("dulieu","id: "+id +" name:"+name+" phone:"+phone);

}

}**catch**(Exception e)

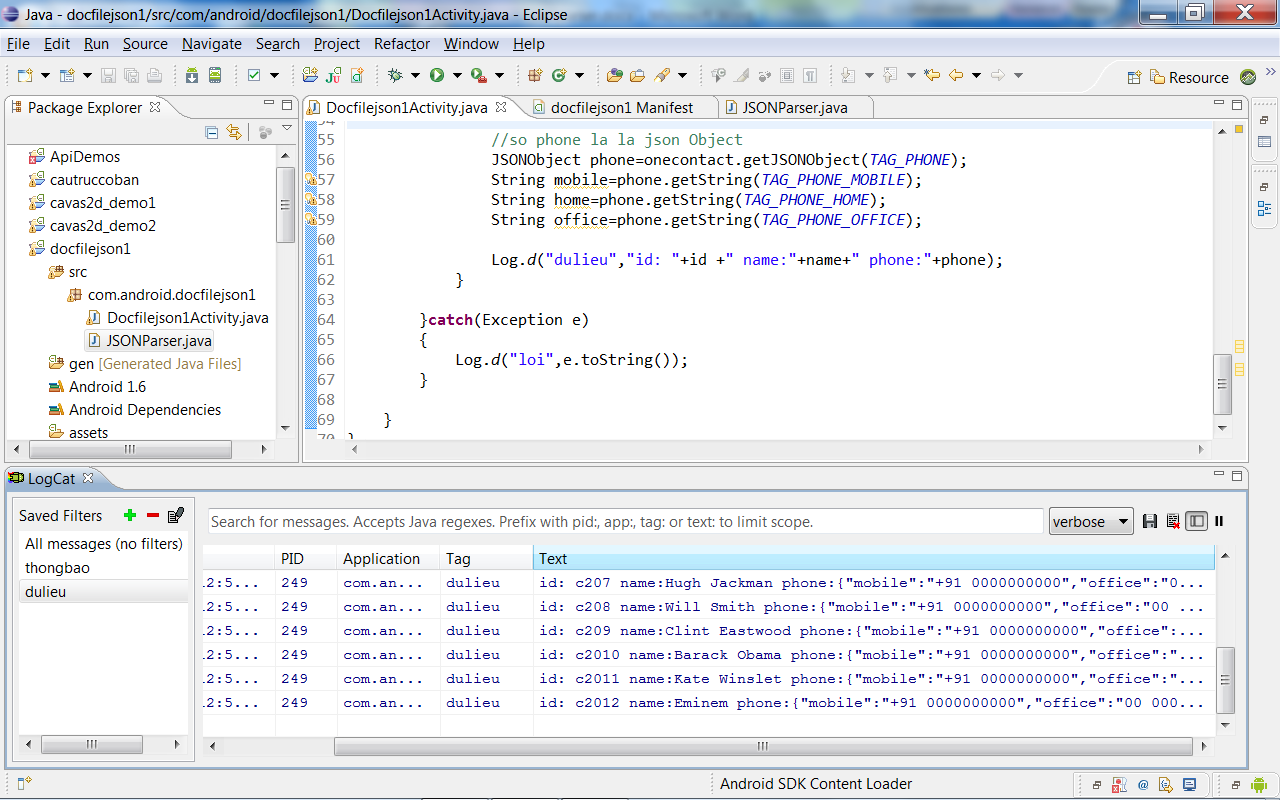
{

Log.*d*("loi",e.toString());

}

}

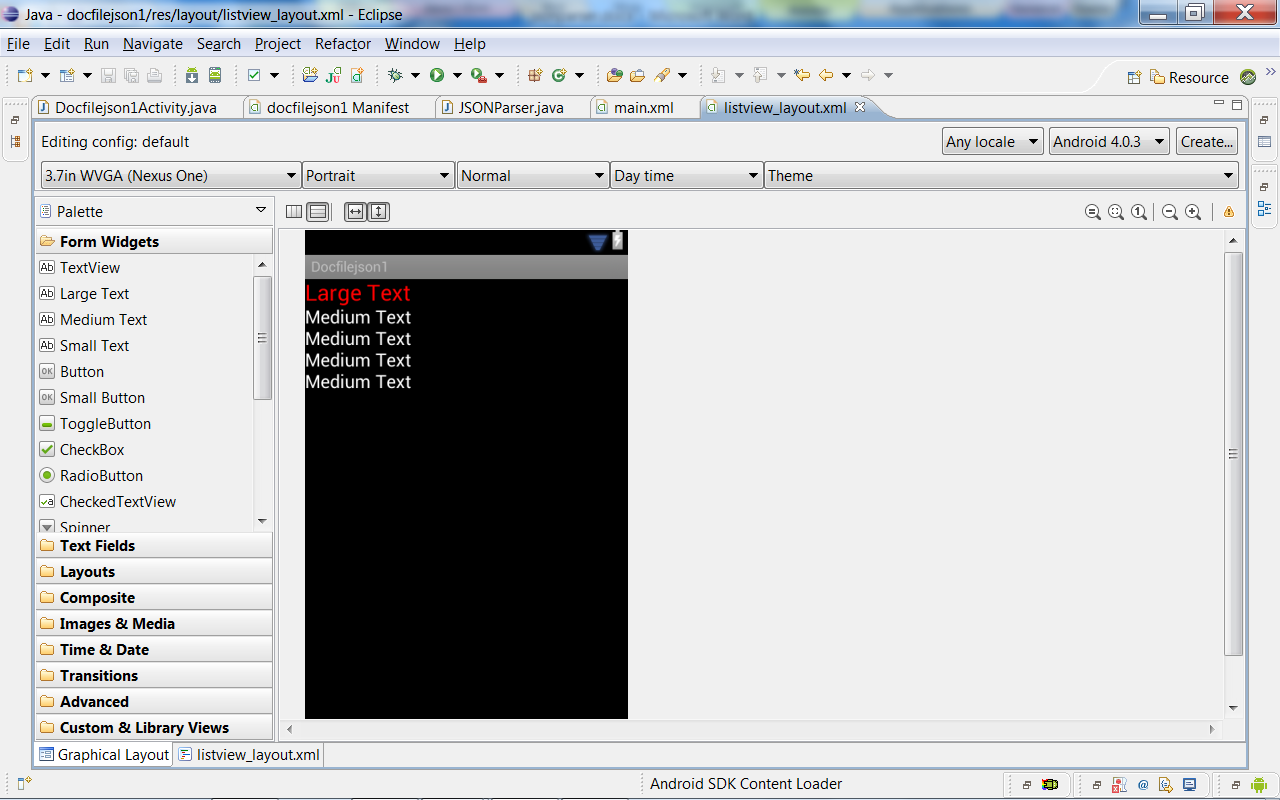
1. Chạy và xem kết quả trong Logcat để thấy dữ liệu JSON đã được phân tích.



# Bài 2: Update bài 1 lên version cao hơn, đưa dữ liệu vào ListView (dùng AsyncTask).

Hướng dẫn.

1. Nâng cấp version lên vd: 4.0.3. Tự làm nhé.
2. Trên giao diện chính kéo vào 1 ListView.
3. Tạo ra 1 layout mới đặt tên “listviewlayout.xml”. Kéo vào 5 TextView đặt tên tuần tự là “name”,”email”, “mobile”, “home”, “office”.



1. Trong file java chính khai báo ListView và ánh xạ nó. Khai báo thêm một ArrayList chứa các HashMap

ListView lv;

ArrayList<HashMap<String,String>> contactlist= **new** ArrayList<HashMap<String,String>>();

1. Xây dựng 1 class trong file java chính (chú ý: AsycnTask phải là subclass) tên ChayNenLayJson kế thừa từ AsyncTask. Override lên 2 hàm là doInBackGround và onPostExecute.

**class** ChayNenLayJson **extends** AsyncTask<Void,Void,Void>

{

@Override

**protected** Void doInBackground(Void... arg0) {

// **TODO** Auto-generated method stub

**return** **null**;

}

@Override

**protected** **void** onPostExecute(Void result) {

// **TODO** Auto-generated method stub

**super**.onPostExecute(result);

}

}

1. Trong hàm onCreate viết lệnh để chạy AsyncTask

**new** ChayNenLayJson().execute();

1. Cut toàn bộ đoạn truy xuất JSON và paste vào hàm doInBackGround. Sau lệnh in ra Logcat thêm lệnh để tạo đối tượng HashMap, lấy các dữ liệu bỏ vào HashMap cuối cùng add HashMap vào ArrayList. Đến đây ta đã có 1 ArrayList chứa tập hợp các dữ liệu đã phân tích được từ Json.

JSONParser jsonparse=**new** JSONParser();

//lay doi tuong json

JSONObject jsonobject=jsonparse.getJSONFromUrl(*url*);

**try**{

//tao mang contacts

contacts=jsonobject.getJSONArray(*TAG\_CONTACTS*);

//duyet tung contacts

**for**(**int** i=0;i<contacts.length();i++)

{

…………………………………………………….

String office=phone.getString(*TAG\_PHONE\_OFFICE*);

Log.*d*("dulieu","id: "+id +" name:"+name+" phone:"+phone);

HashMap<String,String> map=**new** HashMap<String,String>();

map.put(*TAG\_ID*, id);

map.put(*TAG\_NAME*, name);

map.put(*TAG\_EMAIL*,email);

map.put(*TAG\_ADDRESS*,address);

map.put(*TAG\_GENDER*,gender);

map.put(*TAG\_PHONE\_MOBILE*,mobile);

map.put(*TAG\_PHONE\_HOME*,home);

map.put(*TAG\_PHONE\_OFFICE*,office);

contactlist.add(map);

}

}**catch**(Exception e)

{

Log.*d*("loi",e.toString());

}

1. Trong hàm onPostExecute viết thêm lệnh để lấy dữ liệu từ ArrayList và đổ lên ListView.

ListAdapter adapter=**new** SimpleAdapter(getApplicationContext(),

contactlist,

R.layout.*listview\_layout*,

**new** String[]{*TAG\_NAME*,*TAG\_EMAIL*,*TAG\_PHONE\_MOBILE*,*TAG\_PHONE\_HOME*,*TAG\_PHONE\_OFFICE*},

**new** **int**[]{R.id.*name*,R.id.*email*,R.id.*mobile*,R.id.*home*,R.id.*office*});

lv.setAdapter(adapter);

1. Cuối cùng trong hàm onCreate bắt sự kiện khi một mục trong ListView được chọn

lv.setOnItemClickListener(**new** OnItemClickListener() {

**public** **void** onItemClick(AdapterView<?> arg0, View arg1, **int** arg2,

**long** arg3) {

// **TODO** Auto-generated method stub

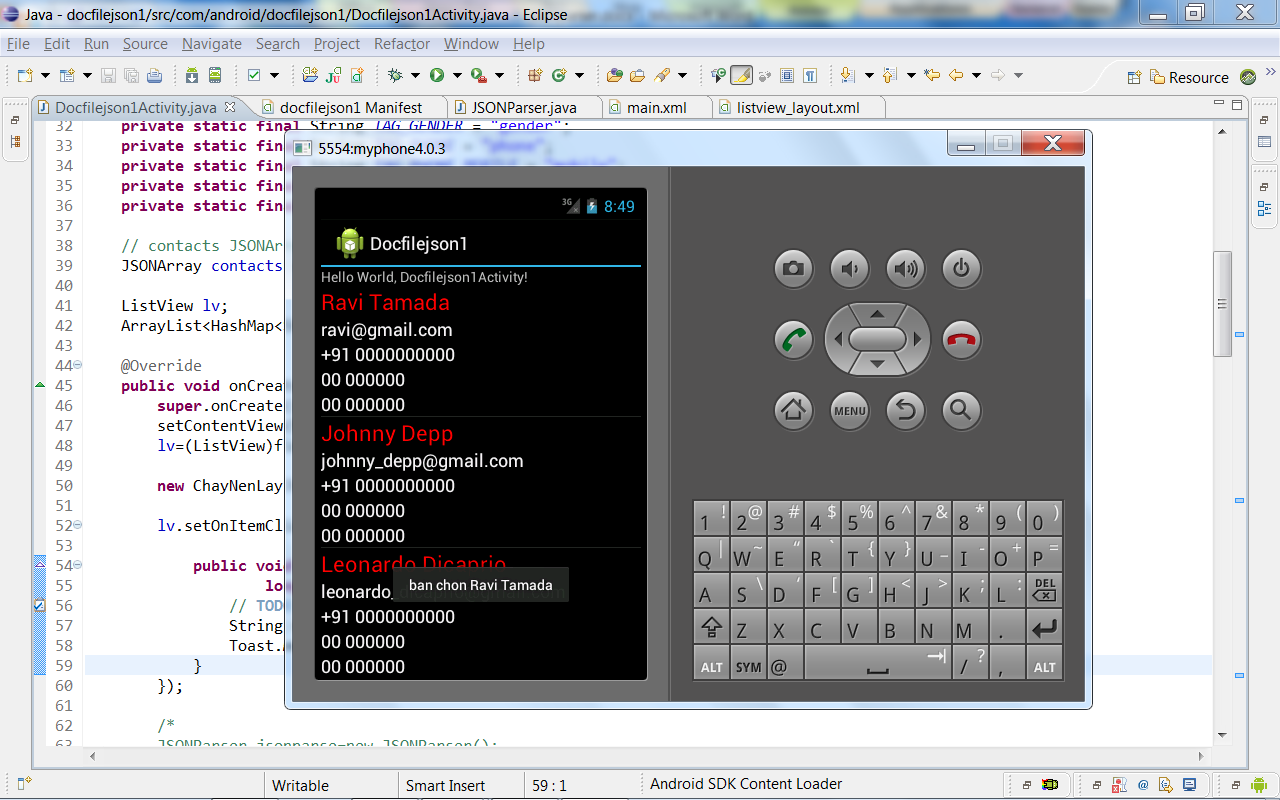
String name=((TextView)arg1.findViewById(R.id.*name*)).getText().toString();

Toast.*makeText*(getApplicationContext(),"ban chon "+name, Toast.*LENGTH\_SHORT*).show();

}

});

1. Chạy chương trình để thấy dữ liệu đã được đổ lên ListView. Click vào một mục trên ListView để thấy Toast.



# Bài 3: Sửa bài 2 và dùng Handler để điều khiển Thread và update UI. Có thể phát triển thêm để khi click vào một mục trên ListView sẽ mở Activity mới và truyền dữ liệu qua.