**455-555 Week 5 Session Notes**

**Chapter Readings:**

Chapter 7 – Async processing, JSON data (including Powerpoint & demo files!)

**Topics:**

UI Implementation : ListViews & various adapters

[JSON](https://developer.android.com/reference/android/util/JsonReader#parsing-json) IO / parsing

Final Project Ideas

Lab 2 review (working a customized adapter – see Week 4 – ListView demo)

Lab 3 overviews

Review 4 on Intents now loaded

­­­­­

**[ Using an ArrayAdapter with ListView ]**

In Android development, any time we want to show a vertical list of scrollable items we will use a ListView which has data populated using an Adapter. The simplest adapter to use is called an ArrayAdapter because the adapter converts an ArrayList of objects into View items loaded into the ListView container.

Diagram

Description automatically generated

The ArrayAdapter fits in between an ArrayList (data source) and the ListView (visual representation) and configures two aspects:

* Which array to use as the data source for the list
* How to convert any given item in the array into a corresponding View object

**[ JSON parsing in Android ]**

[JSON](http://www.json.org/) is short for **JavaScript Object Notation**. JSON is used for data exchange between a server and Android for instance and it is a very condensed data exchange format. JSON is used for storing data in files and it is a very easy method to parse data. Android provides support to parse JSON objects and arrays.

Advantages of JSON over XML  
JSON is faster than XML.  
  
It uses arrays.  
  
JSON is very easy to read and write**.**

Ex. JSON data retrieval (getting data from the Web Server)

A screenshot of a cell phone

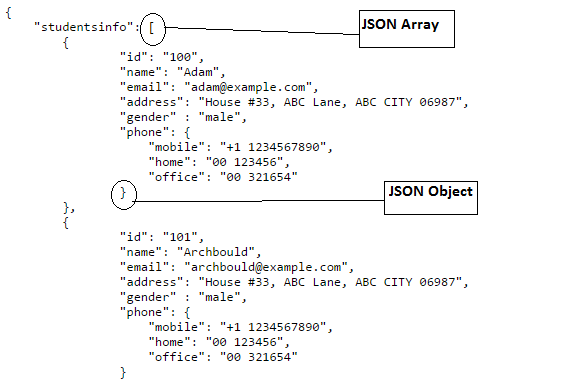
Description automatically generated

Data structures in JSON are based on key / value pairs and we can summarize in these points:

* JSON Object : { string : value , .... }
* JSON Array : [ value , value .....]
* value : string || number || object || array || true || false, || null

If you observe normally JSON data will have square brackets and curly brackets. The difference between **[** and **{** is, the square bracket represents starting of an JSONArray node whereas curly bracket represents JSONObject.

Sample json format:



JSON example:

{

"id":10,

"name":"myname",

"listMessages":

[

{

"value":1,

"text":"value",

"id\_message":0

},

{ "value":11,"text":"value","id\_message":1}

]

}

Android includes the [**json.org**](http://developer.android.com/reference/org/json/JSONObject.html) **libraries** which allow to work easily with JSON files.

**Create a JSONObject:**

JSONObject jsonObject = new JSONObject();

**To put a specific value**

jsonObject.put("id", 10);

jsonObject.put("name", "myname");

**To create a specific array**

JSONArray jsonArrayMessages = new JSONArray();

JSONObject jsonMessage = new JSONObject();

jsonMessage.put("id\_message", 0);

jsonMessage.put("text", "value");

jsonArrayMessages.put(jsonMessage);

**To put a specific array in a JSONObject**

jsonObject.put("listMessages", jsonArrayMessages);

**Create a JSONObject from a JSON String:**

JSONObject jsonObject = new JSONObject(result);

**To get a specific string from JSON Object**

String name = jsonObject.getString("name");

**To get a specific int**

int id = jsonObject.getInt("id");

**To get a specific array**

JSONArray jArray = jsonObject.getJSONArray("listMessages");

**To get the items from the array**

JSONObject msg = jArray.getJSONObject(1);

int id\_message = msg.getInt("id\_message");

**Ex. JSON app example follows- Create objects and display outcomes**

**Imports needed!**

**import** org.json.JSONArray;  
**import** org.json.JSONException;  
**import** org.json.JSONObject;

**Create array with object data for class**

String **strJson** = **"{ \"Employee\" "** +  
 **":[{\"id\":\"101\",\"name\":\"James Papa\",\"salary\":\"50000\"},"** +  
 **"{\"id\":\"102\",\"name\":\"Billy Preston\",\"salary\":\"60000\"}, "** +  
 **"{\"id\":\"103\",\"name\":\"Robin Hood\",\"salary\":\"70000\"}, "** +

**::**  
 **"{\"id\":\"108\",\"name\":\"Timothy VonCina\",\"salary\":\"0\"}] }"**;

**onCreate() method detail**

**try** {  
 *// Create the root JSONObject from the JSON string.* JSONObject jsonRootObject = **new** JSONObject(**strJson**);  
  
 *//Get the instance of JSONArray that contains JSONObjects* JSONArray jsonArray = jsonRootObject.optJSONArray(**"Employee"**);  
  
 ArrayList<JSONObject> listItems = getArrayListFromJSONArray(jsonArray);  
  
 **listV** = findViewById(R.id.***listv***);  
 ArrayAdapter<JSONObject> adapter = new ArrayAdapter<>(this,   
 R.layout.*list\_layout*, R.id.*txtId*, listItems);

**listV**.setAdapter(adapter);  
  
} **catch** (JSONException e) {  
 e.printStackTrace();  
}

**private** ArrayList<JSONObject> getArrayListFromJSONArray(JSONArray jsonArray) {  
  
 ArrayList<JSONObject> aList = **new** ArrayList<JSONObject>();  
  
 **try** {  
 **if** (jsonArray != **null**) {  
 **for** (**int** i = 0; i < jsonArray.length(); i++) {  
 aList.add(jsonArray.getJSONObject(i));  
 }  
 }  
  
 } **catch** (JSONException je) {  
 je.printStackTrace();  
 }  
  
 **return** aList;  
}

**Snapshot follows showing the binded listItems array to the list view.**

**A screenshot of a cell phone

Description automatically generated**

**Challenge. To incorporate the above clean layout view!!**

A screenshot of a cell phone

Description automatically generated*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="horizontal"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background ="@color/teal"  
 android:padding="10dp"**>  
  
 <**TextView  
 android:id="@+id/txtId"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textSize="18dp"  
 android:textStyle="bold"  
 android:textColor="#660C5C"  
 android:layout\_marginTop="0dp"  
 android:layout\_weight="1"** />  
  
 <**TextView  
 android:id="@+id/txtName"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textColor="#000"  
 android:layout\_below="@+id/txtId"  
 android:layout\_alignLeft="@+id/txtId"  
 android:layout\_alignStart="@+id/txtId"  
 android:layout\_marginTop="20dp"  
 android:layout\_weight="1"** />  
  
 <**TextView  
 android:id="@+id/txtSalary"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textColor="#000"  
 android:layout\_alignTop="@+id/txtName"  
 android:layout\_alignParentRight="true"  
 android:layout\_weight="1"** />  
  
</**RelativeLayout**>

Sample LogCat output file

A screenshot of a cell phone

Description automatically generated

**Displaying file in Device File Explorer…**

**While app is in run mode, click the Explorer icon on right hand side**

Text

Description automatically generated



**You’ll notice a pop-up view of your running emulator now appears as**

**shown next.**

**Drill down to the data > data folder. Notice the permissions, date and time stamp and byte size is listed in Linux long format mode.**

A screenshot of a cell phone

Description automatically generated

**With the data > data folder open, drill further down till you see your package and then your files folder which should contain your data file (ex. samplefile.txt). Open file and observe the output.**

Graphical user interface, text, application

Description automatically generated

**And voila! Data now appears!**

A screenshot of a cell phone

Description automatically generated

Final Project ideas (Concepts)

* [iClicker](https://www.iclicker.com/?mkt_tok=eyJpIjoiWlRaak5EZ3pNVGd4T1dFNSIsInQiOiJPaitXYk9LNWZuVTdCd1wvUmlla1ZaRFlYOG12azM2a3FXbE5vUGdvMHhOaFFINVFzck5SWHNZWThabTl0djBCejhwd0pxZ3NQbmYxdkFJTUg2Qlp3OVpkUGRMTkxFSDVkd3o4SlNERm5RZUpSN0dWMVF0cFlSclpiMlpZdlBTUzMifQ%3D%3D)
* [AirPods](https://www.itprotoday.com/ios/new-apple-airpods-may-help-keep-workers-healthy?NL=TECH-001&Issue=TECH-001_20190219_TECH-001_718&sfvc4enews=42&cl=article_1_b&utm_rid=CPNET000014582888&utm_campaign=11071&utm_medium=email&elq2=e1909a8f0819442099920909ac882f67) (Sensor driven!)
* [Data sources](https://www.kaggle.com/lava18/google-play-store-apps#googleplaystore.csv)
* Venue bookings
* UBER Ride Share
* Various types of calculator programs
* Receipt Scanner
* Scan a product & sell it
* Text <--> Speech (Some objectives)
* [Hire me now app](https://play.google.com/store/apps/details?id=com.instaworkmobile&hl=en) (upload resume, vid of self, book interview!)

[ Lab 2 touch points]

Set up a **back** button perhaps (lab 2)

<ImageButton

*<!— use dp’s for layout height / width -->*

android:id="@+id/imageButton"  
 android:layout\_width="52dp"  
 android:layout\_height="46dp"  
 **android:onClick="onClick"**  
 **android:src=**"@drawable/back"

/>

Now do the action!

**public void** onClick(View view) {

TextView textView2 = findViewById(R.id.***textview2***);

**checkBox**.setVisibility(View.***VISIBLE***);  
 **seekBar**.setVisibility(View.***VISIBLE***);  
 **stub**.setVisibility(View.***GONE***);  
 textView2.setVisibility(View.***GONE***);  
 **textView**.setVisibility(View.***VISIBLE***);  
  
 **img**.setVisibility(View.***GONE***);

*//also uncheck checkbox!!*

}

Adding degrees to text

Use "**\u00B0**" to represent special ‘degree’ character

So you can concatentate a string temp to the special character

Ex.

“15” + “\u00B0”

Extra: Overriding the ‘physical’ back button on unit.

Override the onBackPressed method and do nothing if you meant to handle the back button on the device. Code this outside of any method within your Activity.

**@Override**

**public void onBackPressed() {**

**// Simply Do noting!**

**}**

Refs:

<https://developer.android.com/develop/ui/views/touch-and-input/drag-drop>