Toon Land (JSON Async Reads)

Create a new project named *ToonLand*.

Add the following permission to AndroidManifest.xml just before the *<application..>* element:

1. user-permission.txt

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

/drawable

Add *placeholder.png* into the *drawable* folder.

Make layout in *activity\_main.xml* based on *LinearLayout* with *vertical* orientation.

The only control in activity\_main.xml is a *ListView* as shown below:

<ListView

2. ListView.txt

android:id="@+id/toonList"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content" />

Create another layout file named *list\_row\_layout.xml* to render a single list item view with following content:

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="horizontal"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:minHeight="50dp"

android:padding="8dp">

<ImageView

android:id="@+id/thumbImage"

android:layout\_alignParentLeft="true"

android:layout\_centerVertical="true"

android:layout\_width="70dp"

3. list\_row\_layout.xml.txt

android:layout\_height="70dp"

android:layout\_marginRight="10dp" />

<TextView

android:id="@+id/firstName"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginRight="10dp" />

<TextView

android:id="@+id/lastName"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"/>

</LinearLayout>

Create a class named *HttpHandler.java* that is responsible for reading data from the internet. Add to it the following code:

public class HttpHandler {

private static final String TAG = HttpHandler.class.getSimpleName();

public HttpHandler() { }

public String makeServiceCall(String reqUrl) {

4. HttpHandler.java.txt

String response = null;

try {

URL url = new URL(reqUrl);

HttpURLConnection conn = (HttpURLConnection) url.openConnection();

conn.setRequestMethod("GET");

// read the response

InputStream in = new BufferedInputStream(conn.getInputStream());

response = convertStreamToString(in);

} catch (MalformedURLException e) {

Log.e(TAG, "MalformedURLException: " + e.getMessage());

} catch (ProtocolException e) {

Log.e(TAG, "ProtocolException: " + e.getMessage());

} catch (IOException e) {

Log.e(TAG, "IOException: " + e.getMessage());

} catch (Exception e) {

Log.e(TAG, "Exception: " + e.getMessage());

}

return response;

}

private String convertStreamToString(InputStream is) {

BufferedReader reader = new BufferedReader(new InputStreamReader(is));

StringBuilder sb = new StringBuilder();

String line;

try {

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

sb.append(line).append('\n');

}

} catch (IOException e) {

e.printStackTrace();

} finally {

try {

is.close();

} catch (IOException e) {

e.printStackTrace();

}

}

return sb.toString();

}

}

Go to <http://flintstones.zift.ca/api/flintstones/> and copy one record into the notepad:

{

"$id":"1",

"PersonId":1,

"FirstName":"Fred",

"LastName":"Flintstone",

"Occupation":"Mining Manager",

"Gender":"M",

"Created":"2017-10-01T16:54:45.887",

"Picture":"http://flintstones.zift.ca/images/flintstone/fred.png"

}

Delete *"$id":"1",*

Go to <http://www.freecodeformat.com/json2pojo.php>. Paste the single record into the “*Enter JSON:*” then click on the *Generate* button. Create a simple *Toon.java* POJO class with the generated Java code.

Add the following instance variables to *MainActivity.java*.

private String TAG = MainActivity.class.getSimpleName();

5. MainActivity\_instance\_vars.txt

private ProgressDialog pDialog;

private ListView lv;

// URL to get contacts JSON

private static String SERVICE\_URL = "http://flintstones.zift.ca/api/flintstones/";

private ArrayList<Toon> toonList;

We need a service that is capable of asynchronously downloading images from the internet and converting them into bitmaps. This service will be performed by a class named *ImageDownloaderTask* with the following code:

class ImageDownloaderTask extends AsyncTask<String, Void, Bitmap> {

private final WeakReference<ImageView> imageViewReference;

public ImageDownloaderTask(ImageView imageView) {

imageViewReference = new WeakReference<ImageView>(imageView);

}

@Override

protected Bitmap doInBackground(String... params) {

return downloadBitmap(params[0]);

}

6. ImageDownloaderTask.txt

@Override

protected void onPostExecute(Bitmap bitmap) {

if (isCancelled()) {

bitmap = null;

}

if (imageViewReference != null) {

ImageView imageView = imageViewReference.get();

if (imageView != null) {

if (bitmap != null) {

imageView.setImageBitmap(bitmap);

} else {

Drawable placeholder = imageView.getContext().getResources().getDrawable(R.drawable.placeholder);

imageView.setImageDrawable(placeholder);

}

}

}

}

private Bitmap downloadBitmap(String url) {

HttpURLConnection urlConnection = null;

try {

URL uri = new URL(url);

urlConnection = (HttpURLConnection) uri.openConnection();

int statusCode = urlConnection.getResponseCode();

if (statusCode != HttpURLConnection.HTTP\_OK) {

return null;

}

InputStream inputStream = urlConnection.getInputStream();

if (inputStream != null) {

Bitmap bitmap = BitmapFactory.decodeStream(inputStream);

return bitmap;

}

} catch (Exception e) {

urlConnection.disconnect();

Log.w("ImageDownloader", "Error downloading image from " + url);

} finally {

if (urlConnection != null) {

urlConnection.disconnect();

}

}

return null;

}

}

The adapter is the arbitrator between the *Data Source* and the *ListView*. Add the following *ToonsAdapter* with the following code:

public class ToonsAdapter extends ArrayAdapter<Toon> {

Context \_context;

public ToonsAdapter(Context context, ArrayList<Toon> toons) {

super(context, 0, toons);

7. ToonsAdapter.txt

\_context = context;

}

@Override

public View getView(int position, View convertView, ViewGroup parent) {

final Activity activity = (Activity) \_context;

// Get the data item for this position

Toon toon = getItem(position);

// Check if an existing view is being reused, otherwise inflate the view

if (convertView == null) {

convertView = LayoutInflater.from(getContext()).inflate(R.layout.list\_row\_layout, parent, false);

}

// Lookup view for data population

TextView tvFirstName = (TextView) convertView.findViewById(R.id.firstName);

TextView tvLastName = (TextView) convertView.findViewById(R.id.lastName);

// Populate the data into the template view using the data object

tvFirstName.setText(toon.getFirstName());

tvLastName.setText(toon.getLastName());

ImageView imgOnePhoto = (ImageView) convertView.findViewById(R.id.thumbImage);

//DownloadImageTask dit = new DownloadImageTask(\_context, imgOnePhoto);

//dit.execute(toon.getPicture());

if (toon.getPicture() != null) {

new ImageDownloaderTask(imgOnePhoto).execute(toon.getPicture());

}

// Return the completed view to render on screen

return convertView;

}

}

Add this inner class just below the *onCreate()* method in *MainActivity*.

/\*\*

\* Async task class to get json by making HTTP call

\*/

private class GetContacts extends AsyncTask<Void, Void, Void> {

@Override

protected void onPreExecute() {

super.onPreExecute();

// Showing progress dialog

pDialog = new ProgressDialog(MainActivity.this);

pDialog.setMessage("Please wait...");

pDialog.setCancelable(false);

pDialog.show();

8. GetContacts.txt

}

@Override

protected Void doInBackground(Void... arg0) {

HttpHandler sh = new HttpHandler();

// Making a request to url and getting response

String jsonStr = sh.makeServiceCall(SERVICE\_URL);

Log.e(TAG, "Response from url: " + jsonStr);

if (jsonStr != null) {

try {

//JSONObject jsonObj = new JSONObject(jsonStr);

// Getting JSON Array node

JSONArray toonJsonArray = new JSONArray(jsonStr);

// looping through All Contacts

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

JSONObject c = toonJsonArray.getJSONObject(i);

String personId = c.getString("PersonId");

String firstName = c.getString("FirstName");

String lastName = c.getString("LastName");

String occupation = c.getString("Occupation");

String gender = c.getString("Gender");

String created = c.getString("Created");

String picture = c.getString("Picture");

// tmp hash map for single contact

Toon toon = new Toon();

// adding each child node to HashMap key => value

toon.setPersonId(Integer.parseInt(personId));

toon.setFirstName(firstName);

toon.setLastName(lastName);

toon.setOccupation(occupation);

toon.setGender(gender);

toon.setCreated(created);

toon.setPicture(picture);

// adding contact to contact list

toonList.add(toon);

}

} catch (final JSONException e) {

Log.e(TAG, "Json parsing error: " + e.getMessage());

runOnUiThread(new Runnable() {

@Override

public void run() {

Toast.makeText(getApplicationContext(),

"Json parsing error: " + e.getMessage(),

Toast.LENGTH\_LONG)

.show();

}

});

}

} else {

Log.e(TAG, "Couldn't get json from server.");

runOnUiThread(new Runnable() {

@Override

public void run() {

Toast.makeText(getApplicationContext(),

"Couldn't get json from server. Check LogCat for possible errors!",

Toast.LENGTH\_LONG)

.show();

}

});

}

return null;

}

@Override

protected void onPostExecute(Void result) {

super.onPostExecute(result);

// Dismiss the progress dialog

if (pDialog.isShowing())

pDialog.dismiss();

//Toon[] toonArray = toonList.toArray(new Toon[toonList.size()]);

ToonsAdapter adapter = new ToonsAdapter(MainActivity.this, toonList);

// Attach the adapter to a ListView

lv.setAdapter(adapter);

}

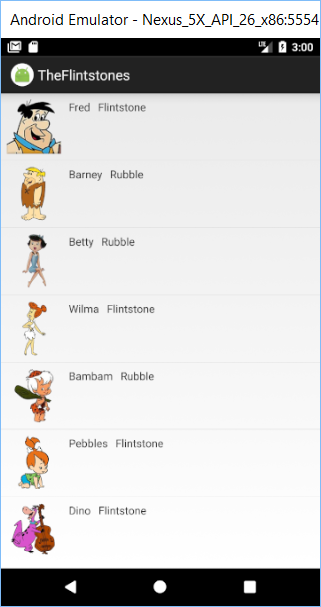
}

Add this code to the *onCreate()* method in *MainActivity*:

9. onCreate.txt

toonList = new ArrayList<Toon>();

lv = (ListView) findViewById(R.id.toonList);

new GetContacts().execute();

Run the app.