Name : Mujahid Ullah

Subject: Android Development

Submitted To Sir Junaid

Date : 31-Dec-20

Assignment No 8

What is Volley?

Volley is an HTTP library that makes networking very **easy and fast**, for Android apps. It was developed by **Google** and introduced during Google I/O 2013. It was developed because there is an absence in Android SDK, of a networking class capable of working without interfering with the user experience. Although Volley is a part of the Android Open Source Project(AOSP), **Google announced in January 2017** that Volley will move to a standalone library. It manages the processing and caching of network requests and it saves developers valuable time from writing the same network call/cache code again and again.

Volley is not suitable **for large download or streaming operations** since Volley holds all responses in memory during parsing.

Features of Volley:

- Request queuing and prioritization
- Effective request cache and memory management
- Extensibility and customization of the library to our needs
- Cancelling the requests

Advantages of using Volley:

- All the task that need to be done with Networking in Android, can be done with the help of Volley.
- Automatic scheduling of network requests.
- Catching
- Multiple concurrent network connections.
- Cancelling request API.
- Request prioritization.
- Volley provides debugging and tracing tools.

How to Import Volley and add Permissions:

Before getting started with Volley, one needs to import Volley and add permissions in the Android Project. **The steps to do so are as follows**:

1. Create new project.
2. Open build.gradle(Module: app) and add the following dependency

```
implementation 'com.android.volley:volley:1.1.1'
```

3. In AndroidManifest.xml add the internet permission:

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

Volley has two main classes:

1. Request Queue: It is the interest one uses for dispatching requests to the network. One can make a request queue on demand if required, but typically it is created early on, at startup time, and keep it around and use it as a Singleton.
2. Request: All the necessary information for making web API call is stored in it. It is the base for creating network requests(GET, POST).

Types of Request using Volley Library:

1) String Request

```
String url = "https:// string_url/";
StringRequest stringRequest = new StringRequest(
        Request.Method.GET,
        url,
        new Response.Listener() {
            @Override
            public void onResponse(String response)
            {
```

```
                }
            },
            new Response.ErrorListener() {
                @Override
                public void onErrorResponse(VolleyError error)
                {
                }
            });
requestQueue.add(stringRequest);
</pre>
```

2) JSONObject Request

```
String url = "https:// json_url/";
    JsonObjectRequest jsonObjectRequest = new JsonObjectRequest(
            Request.Method.GET,
            url,
            null,
            new Response.Listener() {
                @Override
                public void onResponse(JSONObject response)
                {
                }
            },
            new Response.ErrorListener() {
                @Override
                public void onErrorResponse(VolleyError error)
                {
                }
            });
requestQueue.add(jsonObjectRequest);
```

3) JSONArray Request

```
JsonArrayRequest jsonArrayRequest = new JsonArrayRequest(
            Request.Method.GET,
            url,
            null,
            new Response.Listener() {
                @Override
                public void onResponse(JSONArray response)
                {
                }
            },
            new Response.ErrorListener() {
                @Override
                public void onErrorResponse(VolleyError error)
                {
                }
            });
requestQueue.add(jsonArrayRequest);
```

4) Image Request

```
int max - width = ...;
        int max_height = ...;

        String URL = "http:// image_url.png";

        ImageRequest imageRequest= new ImageRequest(URL,
        new Response.Listener() {
@Override
public void
        onResponse(Bitmap response)
        {
        // Assign the response
        // to an ImageView
        ImageView
        imageView
        = (ImageView)
        findViewById(
        R.id.imageView);

        imageView.setImageBitmap(response);
        }
        },
        max_width, max_height, null);

        requestQueue.add(imageRequest);
```

5) Adding Post Parameters

```
    String tag_json_obj = "json_obj_req";

    String url = "https:// api.xyz.info/volley/person_object.json";

    ProgressDialog pDialog = new ProgressDialog(this);
pDialog.setMessage("Loading...PLease wait");
        pDialog.show();

        JsonObjectRequest jsonObjReq= new JsonObjectRequest(
        Method.POST,
        url,
        null,
        new Response.Listener() {

@Override
public void onResponse(JSONObject response)
        {
        Log.d(TAG, response.toString());
        pDialog.hide();
        }
        },
        new Response.ErrorListener() {

@Override
public void onErrorResponse(VolleyError error)
        {
        VolleyLog.d(TAG, "Error: "
        + error.getMessage());
```

```
        pDialog.hide();
        }
        }) {

@Override
protected Map getParams()
        {
        Map params = new HashMap();
        params.put("name", "Mujahid");
        params.put("email", "mujahid.info");
        params.put("password", "password123");

        return params;
        }

        };

        AppController.getInstance()
        .addToRequestQueue(jsonObjReq, tag_json_obj);
```

6) Adding Request Headers

```
String tag_json_obj = "json_obj_req";

    String
            url
            = "https:// api.androidhive.info/volley/person_object.json";

    ProgressDialog pDialog = new ProgressDialog(this);
pDialog.setMessage("Loading...");
        pDialog.show();

        JsonObjectRequest jsonObjReq= new JsonObjectRequest(
        Method.POST,
        url,
        null,
        new Response.Listener() {

@Override
public void onResponse(JSONObject response)
        {
        Log.d(TAG, response.toString());
        pDialog.hide();
        }
        },
        new Response.ErrorListener() {

@Override
public void onErrorResponse(VolleyError error)
        {
        VolleyLog.d(TAG, "Error: "
        + error.getMessage());
        pDialog.hide();
        }
        }) {

@Override
public Map getHeaders() throws AuthFailureError
        {
```

```
        HashMap headers = new HashMap();
        headers.put("Content-Type", "application/json");
        headers.put("apiKey", "xxxxxxxxxxxxxxx");
        return headers;
        }

        };

        AppController.getInstance()
        .addToRequestQueue(jsonObjReq, tag_json_obj);
```

7) Handling the Volley Cache

```
    // Loading request from cache
    Cache
            cache
            = AppController.getInstance()
            .getRequestQueue()
            .getCache();

    Entry entry = cache.get(url);
if (entry != null) {
        try {
        String
        data
        = new String(entry.data, "UTF-8");
        // handle data, like converting it
        // to xml, json, bitmap etc.,
        }
        catch (UnsupportedEncodingException e) {
        e.printStackTrace();
        }
        }
        }
        else
        {
        // If cached response doesn't exists
        }

// Invalidate cache
        AppController.getInstance()
        .getRequestQueue()
        .getCache()
        .invalidate(url, true);

// Turning off cache
// String request
        StringRequest
        stringReq
        = new StringRequest(....);

// disable cache
        stringReq.setShouldCache(false);
```

```
// Deleting cache for particular cache</strong>
        AppController.getInstance()
        .getRequestQueue()
        .getCache()
        .remove(url);

// Deleting all the cache
        AppController.getInstance()
        .getRequestQueue()
        .getCache()
        .clear(url);
```

8) Cancelling Request

```
    // Cancel single request
    String tag_json_arry = "json_req";

ApplicationController.getInstance()
        .getRequestQueue()
        .cancelAll("feed_request");

// Cancel all request
        ApplicationController.getInstance()
        .getRequestQueue()
        .cancelAll();
```

9) Request Prioritization

```
private Priority priority = Priority.HIGH;

StringRequest strReq = new StringRequest(
        Method.GET,
        Const.URL_STRING_REQ,
        new Response
                .Listener() {

            @Override
            public void onResponse(String response) {

                Log.d(TAG, response.toString());
                msgResponse.setText(response.toString());
                hideProgressDialog();

            } },
        new Response
                .ErrorListener() {

            @Override
            public void
            onErrorResponse(VolleyError error) {

                VolleyLog.d(TAG,
                        "Error: "
                                + error.getMessage());
                hideProgressDialog();
            } }) {
```

```
    @Override
    public Priority getPriority()
    {
        return priority;
    }

};
```

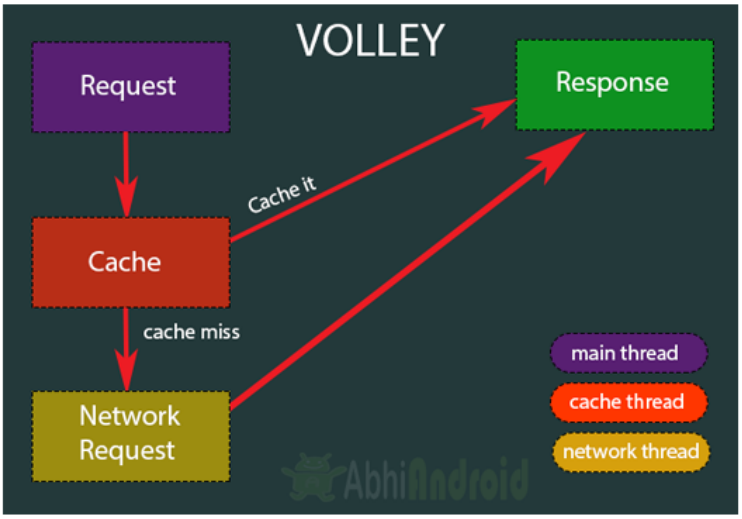
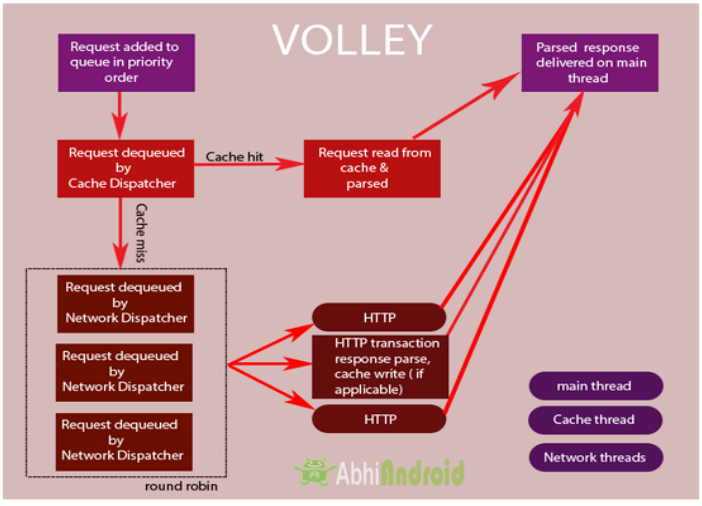


Diagram show how volley improve the performance of application

Request Constructors used in Volley takes 4 parameter:

- First Parameter: Request.Method.GET – The GET is used to read. You can also use POST (to create), PUT (To update/replace), DELETE (to delete), PATCH (to update/modify) and more.
- Second Parameter: URL – The url that will response to the HTTP request.
- Third Parameter: Successful Response Listener – Where your data will go after the request is successfully complete.
- Fourth Parameter: Error Listener – What will be told if there was a problem with your request. For example, you can display it in Log to see the error.



Example Fetch Data From Server by using Volley Library and show in Recycler View

All Steps

1. Create new project.
2. Open build.gradle(Module: app) and add the following dependency

```
implementation 'com.android.volley:volley:1.1.1'
implementation "androidx.recyclerview:recyclerview:1.1.0"
```

3. In AndroidManifest.xml add the internet permission:

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

4. **Create Model class**

5. **create new class name List_data**

```
package com.cal.fetchdatabyusingvolleyrecycler_view;

public class Model_List_data_ {

    private String name;
    private String image_url;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getImage_url() {
        return image_url;
    }

    public void setImage_url(String image_url) {
        this.image_url = image_url;
    }

    public Model_List_data_(String name, String image_url) {
        this.name = name;
        this.image_url = image_url;
    }
}
```

6. **Creating MyAdapter.java class and following the code.**
7.
```
package com.cal.fetchdatabyusingvolleyrecycler_view;

import android.content.Context;
import android.view.LayoutInflater;
```

```
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import com.bumptech.glide.Glide;
import com.squareup.picasso.Picasso;

import java.text.BreakIterator;
import java.util.List;

public class MyAdapter extends RecyclerView.Adapter<MyAdapter.ViewHolder> {

    private List<Model_List_data_>model_list_data_s;
    private Context context;

    public MyAdapter(List<Model_List_data_> model_list_data_s, Context
context) {
        this.model_list_data_s = model_list_data_s;
        this.context = context;
    }

    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {

        View view=
LayoutInflater.from(parent.getContext()).inflate(R.layout.list_data,parent,
false);
        return new ViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull ViewHolder holder, int position)
{

        Model_List_data_ list_data_=model_list_data_s.get(position);

//        Glide
//                .with(context)
//                .load(list_data_.getImage_url())
//                .centerCrop()
//                .into(holder.img);
//        holder.txtname.setText(list_data_.getName());


        Picasso.get().load(list_data_.getImage_url()).into(holder.img);

        holder.txtname.setText(list_data_.getName());



    }
```

```
        @Override
        public int getItemCount() {
            return model_list_data_s.size();
        }

        public class ViewHolder extends RecyclerView.ViewHolder {
            public ImageView img;
            public TextView txtname;

            public ViewHolder(@NonNull View itemView) {
                super(itemView);

                img=(ImageView)itemView.findViewById(R.id.image_view);
                txtname=(TextView)itemView.findViewById(R.id.text_name);



            }
        }
    }
```

7.Main Activity.java Code

```
package com.cal.fetchdatabyusingvolleyrecycler_view;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.JsonArrayRequest;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

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

import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    private static final String HI =
"https://uniqueandrocode.000webhostapp.com/hiren/androidweb.php";
    private RecyclerView rv;
    private List<Model_List_data_>list_data;
    private MyAdapter adapter;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        rv=(RecyclerView)findViewById(R.id.recycleViewContainer);
        rv.setHasFixedSize(true);
        rv.setLayoutManager(new LinearLayoutManager(this));
        list_data=new ArrayList<>();
        adapter=new MyAdapter(list_data,this);




        getData();

    }

    private void getData() {
        StringRequest stringRequest=new StringRequest(Request.Method.GET, HI, new
Response.Listener<String>() {
                    @Override
                    public void onResponse(String response) {

                        Toast.makeText(MainActivity.this, response,
Toast.LENGTH_SHORT).show();
                        try {
                            JSONObject jsonObject=new JSONObject(response);
                            JSONArray jsonArrayd=jsonObject.getJSONArray("data");

                            for (int i=0; i<jsonArrayd.length();i++){
                                JSONObject
jsonObject1=jsonArrayd.getJSONObject(i);

                                Model_List_data_ listData=new
Model_List_data_(jsonObject1.getString("name")
                                        ,jsonObject1.getString("imageurl"));

                                list_data.add(listData);

                                rv.setAdapter(adapter);

                                Button mybtn=findViewById(R.id.btn);

                                mybtn.setOnClickListener(new
View.OnClickListener() {
                                    @Override
                                    public void onClick(View view) {
                                        Toast.makeText(MainActivity.this,
response, Toast.LENGTH_SHORT).show();
                                    }
                                });


                            }



                        } catch (JSONException e) {
                            e.printStackTrace();
                        }
```

```
                    }
                },

                    new Response.ErrorListener() {
                @Override
                public void onErrorResponse(VolleyError error) {

                }
            }

            );
            RequestQueue requestQueue= Volley.newRequestQueue(this);
            requestQueue.add(stringRequest);

        }


    }
```

9) Listdata.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:orientation="vertical"
    android:layout_height="match_parent">
    <ImageView
        android:id="@+id/image_view"
        android:layout_width="match_parent"
        android:layout_height="200dp" />
    <TextView
        android:id="@+id/text_name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="name"
        android:layout_centerHorizontal="true"
        android:layout_margin="10dp"
        android:textSize="20sp"
        android:layout_below="@+id/image_view"
        android:textColor="@android:color/background_dark"/>

</RelativeLayout>
```

10)activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

   <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recycleViewContainer"
        android:layout_width="match_parent"
        android:layout_height="400dp"
```

```
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"/>
    <Button
        android:id="@+id/btn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Click"
        android:layout_below="@+id/recycleViewContainer"
        />

</RelativeLayout>
```