Name: Prashant Bhosale
Roll No: 804

EXPERIMENT 18

WORKING WITH VOLLEY

AIM: To create an application that will fetch and display data from server using volley.

THEORY:

Volley is a HTTP library developed by Google and was first introduced during Google I/O 2013. This library is used to transmit data over the network. It actually makes networking faster and easier for Apps.

The **volley** library has the features like automatic scheduling of network request, multiple concurrent connections, request prioritization, cancel/block a request, easier management of UI with data fetched asynchronously from the network and also offers easier customization.

Volley uses cache to improve the App performance by saving memory and bandwidth of remote server.

Understanding RequestQueue & Working With Volley In Android:

Volley is a networking library managed by the RequestQueue and mainly used for smaller Networking purposes in Android. To use it, first you need to instantiate the RequestQueue and later on you can start or stop request, add or cancel request and access the response cache(s).

```
RequestQueue queue = Volley.newRequestQueue(this);
```

After instantiating RequestQueue, a request must be created. The default request classes already included in Volley library are String request, JSON request, and image request. You can also create custom request by extending Volley's request class.

Request Constructors used in Volley takes 4 parameter:

```
StringRequest request = StringRequest(Request.Method.GET, url, new ResponseListener(), new ErrorListener());
```

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.

Now the last step is to add your request to Request queue and rest volley will handle for you.

```
queue.add(request);
```

Name: Prashant Bhosale
Roll No: 804

ASSIGNMENT

1. Write a basic application to (use volley library), create a button and on click of the button a HTTP request will be send to server. The response from the server is displayed using toast on the screen.

CODE

Import Library : implementation **'com.android.volley:volley:1.0.0'**
Add Internet Permission : <**uses-permission android:name="android.permission.INTERNET"**/>

MainActivity.java

```
public class MainActivity extends AppCompatActivity {

    Button b1;
    private static final String TAG = "VOLLEY ERROR";
    private RequestQueue requestQueue;
    private StringRequest stringRequest;
    private String url = "https://api.myjson.com/bins/x3eok";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        b1 = findViewById(R.id.btn);

        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                sendAndReqRes();
            }
        });
    }

    public void sendAndReqRes(){
        requestQueue = Volley.newRequestQueue(this);
                    stringRequest = new StringRequest(Request.Method.GET, url, new
Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                    Toast.makeText(getApplicationContext(), "Response : "+response.toString(),
Toast.LENGTH_SHORT).show();
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Log.i(TAG, "Error : "+error.toString());
            }
        });
        requestQueue.add(stringRequest);
```

```
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".MainActivity">

   <Button
      android:id="@+id/btn"
      android:text="Click"
      android:textSize="20dp"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content" />
</android.support.constraint.ConstraintLayout>
```

OUTPUT