Volley

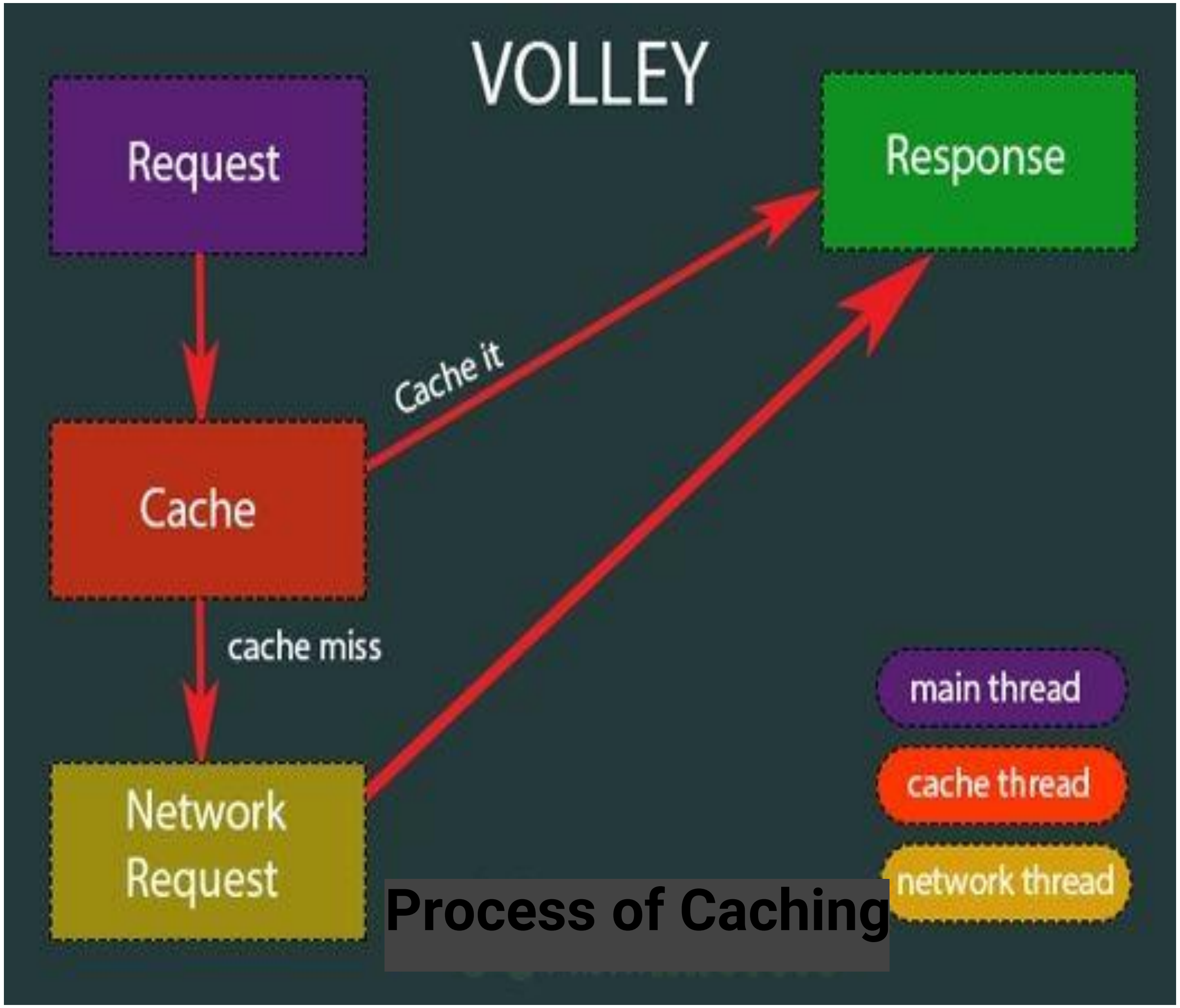
Volley

- 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.
- 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.
- Android volley is a networking library that was introduced to make networking calls much easier, faster without writing tons of code.
- By default all the volley network calls work asynchronously, so we don’t have to worry about using AsyncTask anymore.
- 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.

Volley

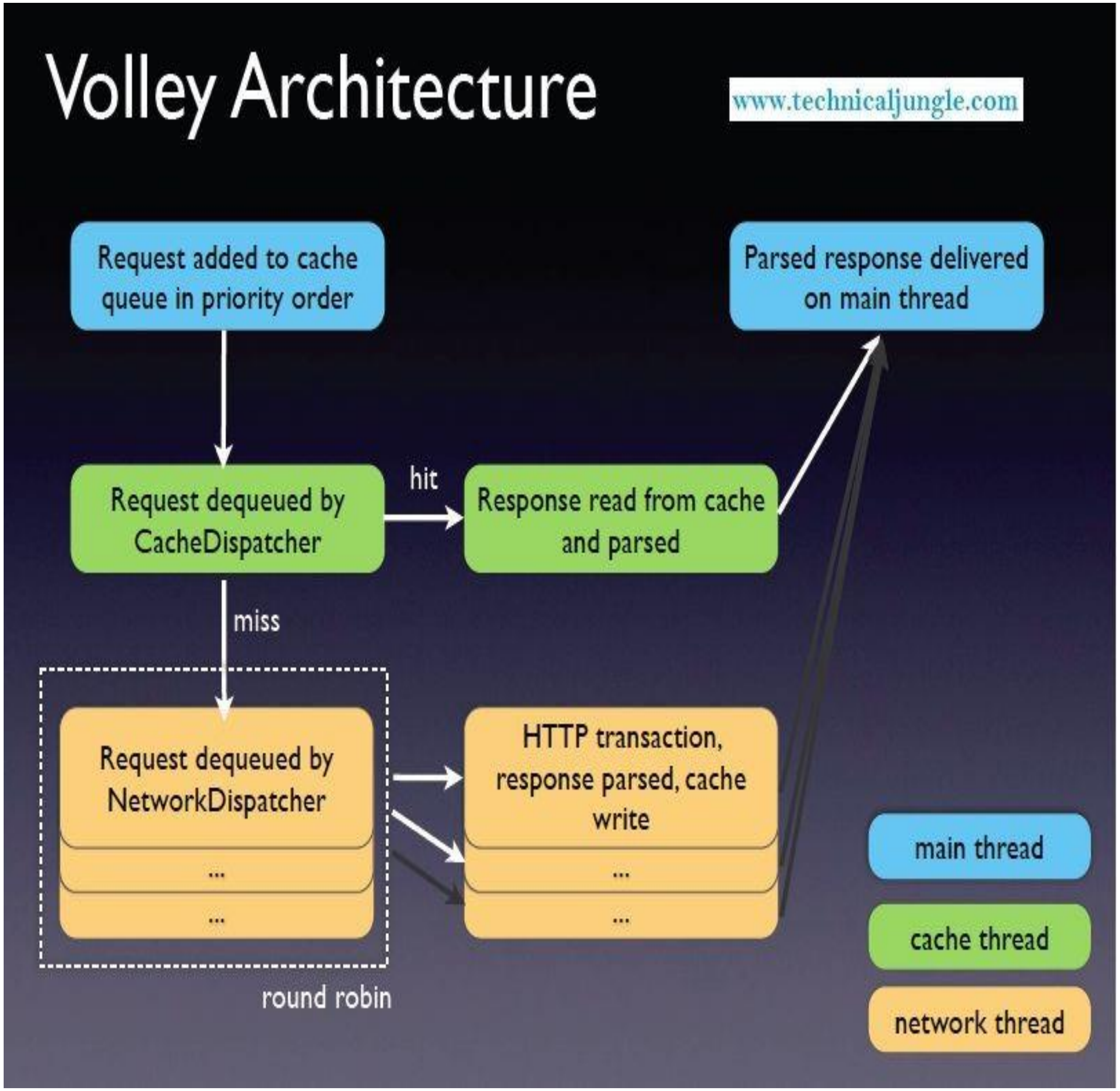
The volley library has the features like automatic scheduling of network request, multiple concurrent connections, request prioritization, cancel/block a request, easier management of UI with data fetched asynchronously from the network and also offers easier customization.

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



Process of Caching

Volley Architecture


www.technicaljungle.com
Request added to cache queue in priority order
Parsed response delivered on main thread
hit
Request dequeued by CacheDispatcher
Response read from cache and parsed
miss
Request dequeued by NetworkDispatcher
...
...
HTTP transaction, response parsed, cache write
...
...
round robin
main thread
cache thread
network thread

Features of Volley

1. Request queuing and prioritization
2. Effective request cache and memory management
3. Extensibility and customization of the library to our needs
4. Cancelling the requests

Advantages of using Volley

1. All the task that need to be done with Networking in Android, can be done with the help of Volley.
2. Automatic scheduling of network requests.
3. Catching
4. Multiple concurrent network connections.
5. Cancelling request API.- You can cancel a single request, or you can set blocks or scopes of requests to cancel.
6. Request prioritization.- Requests will be processed from higher priorities to lower priorities, in FIFO order.
7. 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.Open build.gradle(Module: app) and add the following dependency

```
dependencies{
    //...
    implementation 'com.android.volley:volley:1.0.0'
}
```

2. In AndroidManifest.xml add the internet permission

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

Classes in Volley Library

Volley has two main classes:

1. **Request Queue:** A RequestQueue is used to queue all the requests and handle the responses.

1. **Request:** All the necessary information for making web API call is stored in it. It is the base for creating network requests(GET, POST).

A Request object major types

- JsonObjectRequest

  — To send and receive JSON Object from the server

- JsonArrayRequest

  — To receive JSON Array from the server

- ImageRequest

  -— To receive an image from the server

- StringRequest

  — To retrieve response body as String (ideally if you intend to parse the response by yourself)

JsonObjectRequest

public **JsonObjectRequest**(int method, String url, JSONObject jsonRequest, Response.Listener<JSONObject> listener, Response.ErrorListener errorListener)

- Creates a new request.
- **Parameters**
  - method - the HTTP method to use
  - url - URL to fetch the JSON from
  - jsonRequest - A JSONObject to post with the request. Null is allowed and indicates no parameters will be posted along with request.
  - listener - Listener to receive the JSON response
  - errorListener - Error listener, or null to ignore errors.

Interface Response.Listener<T>

Class Response<T>

- com.android.volley.Response<T>

- **Type Parameters**

  T - Parsed type of this response

public class **Response<T>** extends java.lang.Object

It encapsulates a parsed response for delivery.

Interface Response.Listener<T>

com.android.volley

public static interface Response.Listener<T>

Callback interface for delivering parsed responses.

Method Detail

onResponse

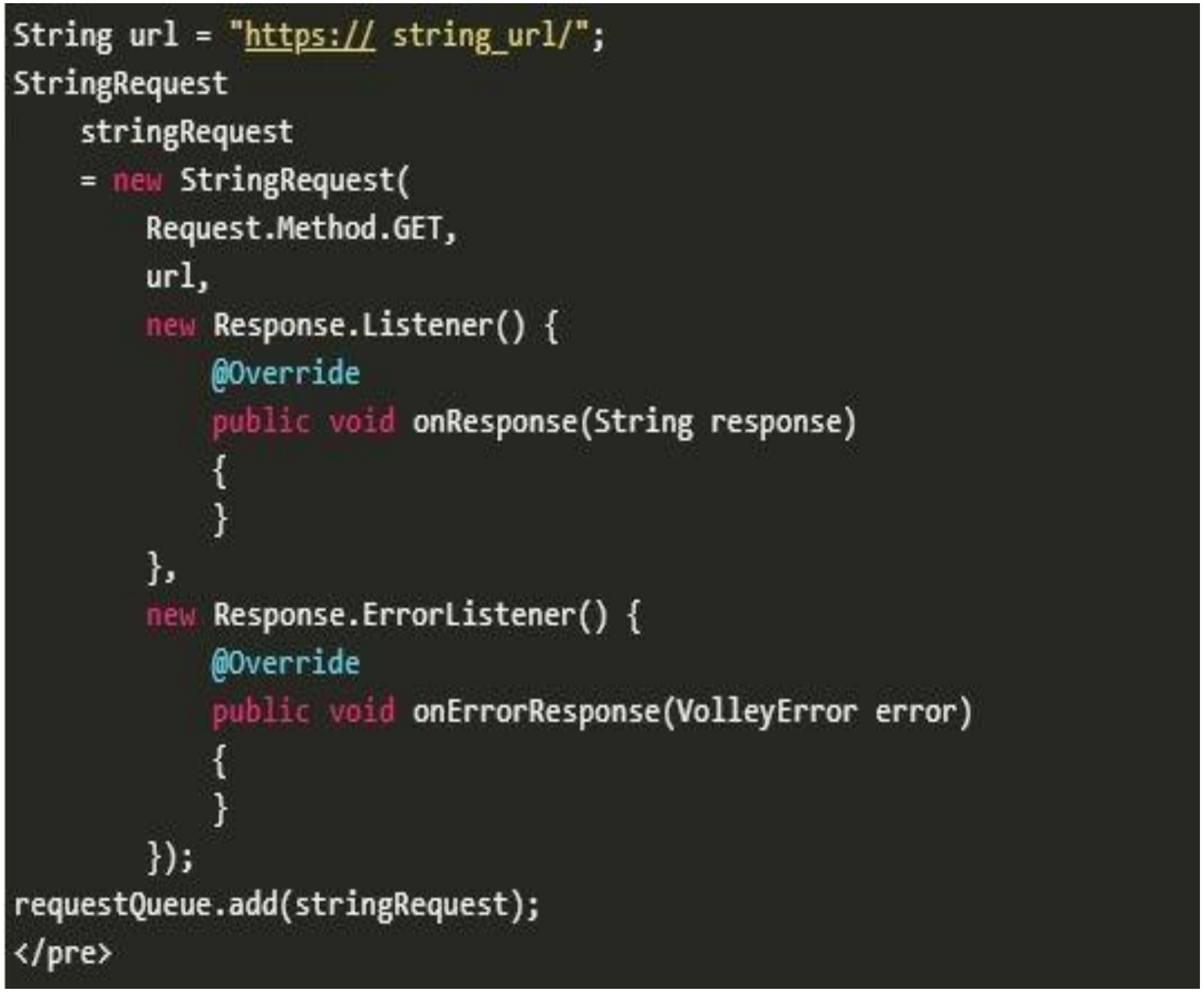
void onResponse(T response)

Called when a response is received.

Interface Response.ErrorListener

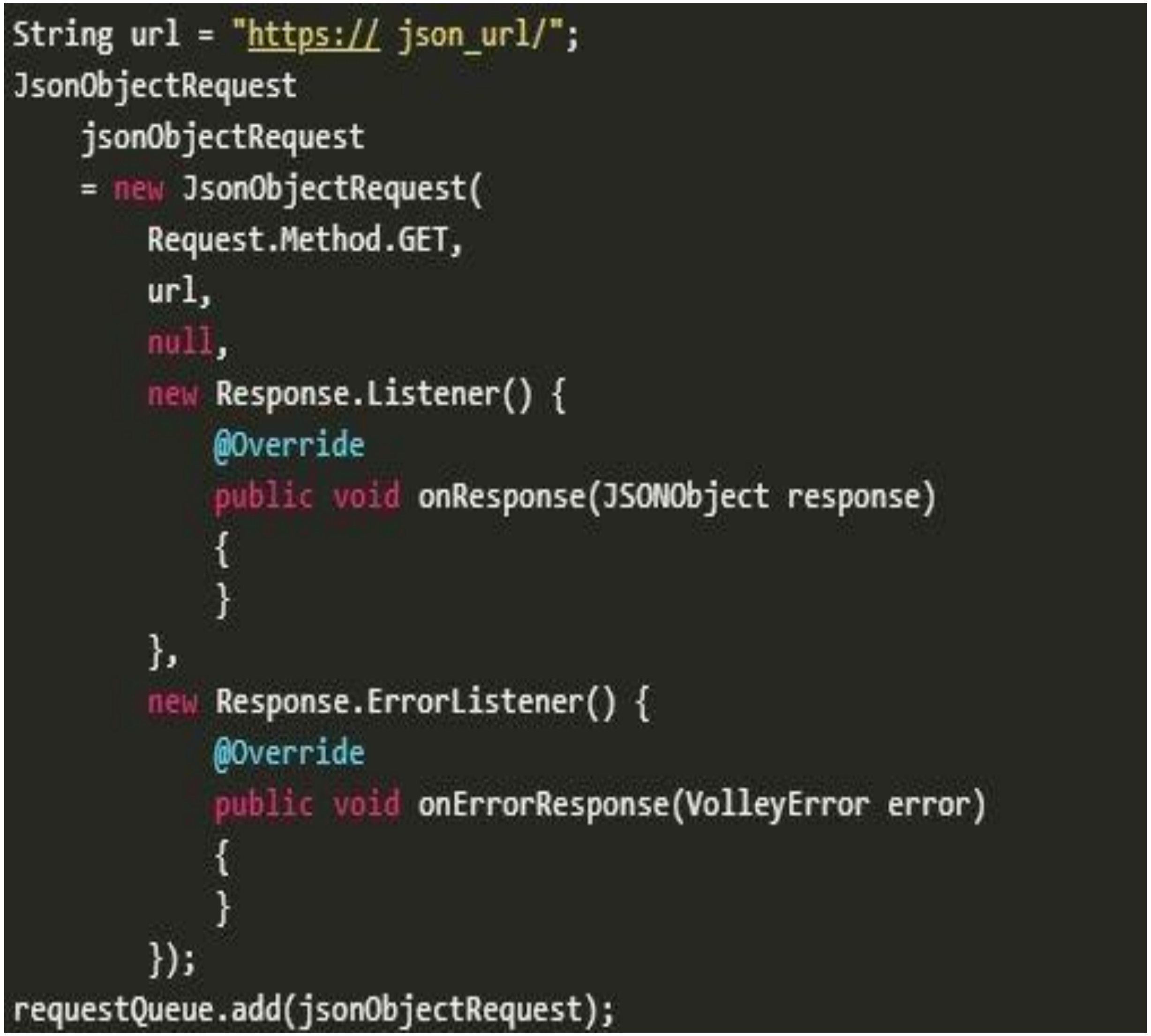
- **Interface Response.ErrorListener**
  - **com.android.volley**
- **Enclosing class**
  - Response<T>
- **Syntax**
  - public static interface **Response.ErrorListener**
- Callback interface for delivering error responses.
- **Method Detail**
  - **onErrorResponse**
  - **Syntax**
    - void **onErrorResponse**(VolleyError error)
  - Callback method that an error has been occurred with the provided error code and optional user-readable message.

Types of Request using Volley Library: String Request



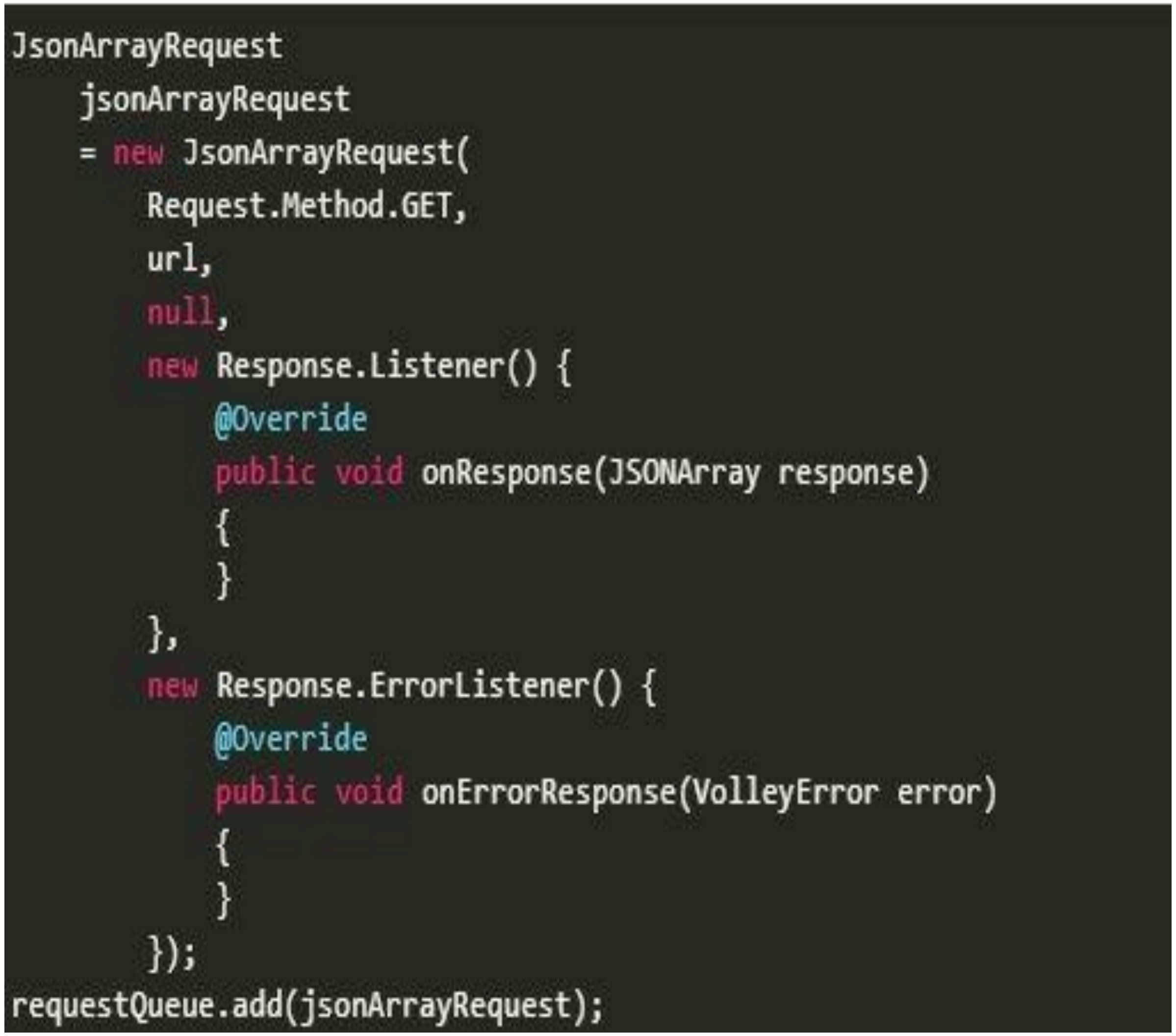
```
String url = "https:// string_url/";
StringRequest
    stringRequest
    = new StringRequest(
        Request.Method.GET,
        url,
        new Response.Listener() {
            @Override
            public void onResponse(String response)
            {
            }
        },
        new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error)
            {
            }
        });
requestQueue.add(stringRequest);
</pre>
```

JSONObject Request



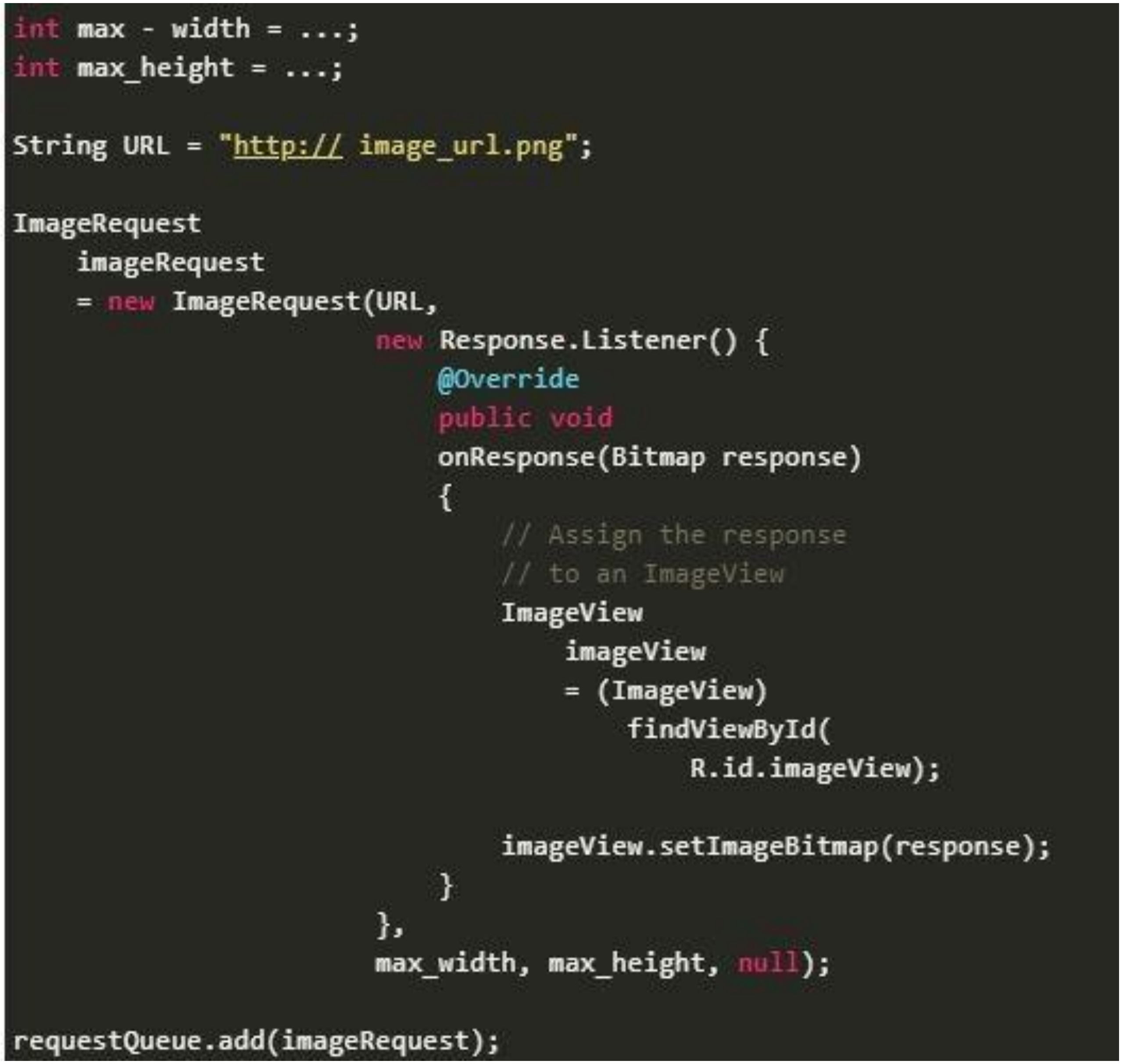
```
String url = "https:// json_url/";
JsonObjectRequest
    jsonObjectRequest
    = new JsonObjectRequest(
        Request.Method.GET,
        url,
        null,
        new Response.Listener() {
            @Override
            public void onResponse(JSONObject response)
            {
           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

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);
```

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);
```

Canceling Requests

```
StringRequest stringRequest = ...;
RequestQueue mRequestQueue = ...;

// Set the tag on the request.
stringRequest.setTag(TAG);

// Add the request to the RequestQueue.
mRequestQueue.add(stringRequest);
```

You can now cancel all requests with this tag using the cancelAll on the request queue:

```
@Override
protected void onStop() {
    super.onStop();
    if (mRequestQueue != null) {
        mRequestQueue.cancelAll(TAG);
    }
}
```

Difference Between Retrofit and Volley In Android

Retrofit is a REST client for Android, through which you can make easy to use interface while Volley is a networking library.

What is Retrofit?

1. Retrofit is a REST client for Android, through which you can make easy to use interfaces which can turn any Android app into a powerful one.
2. It is developed by Square Inc.
3. Retrofit turns your REST API into a Java interface.
4. Retrofit can perform Async and sync requests with automatic JSON parsing without any effort.

What is Volley?

1. Volley is a networking library it offers great features like synchronous requests, asynchronous requests, prioritization, making multiple requests at the same time, ordered requests, and of course caching.

Android & Web services

Which one to use?

	Retrofit	Volley
Maintained by	Square	Google
Coding Complexity	Brief	Boilerplate code
Performance	Quickest	Quick
Caching Support	Modified okHttp	Elaborate support
Image Operations	Add Picasso	Natively supported
Retry Policy	No Default Retry	Natively supported

- **Create an android application to demonstrate JSON data parsing using Volley (you can use https://api.github.com/users json data).**

build.gradle(:app)

```
plugins {
    id 'com.android.application'
}

android {
    compileSdk 31

    defaultConfig {
        applicationId "com.example.jasondataparsingvolley"
        minSdk 21
        targetSdk 31
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'),
'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {
    implementation 'com.android.volley:volley:1.2.1'
    implementation 'com.squareup.picasso:picasso:2.71828'
    implementation 'androidx.appcompat:appcompat:1.4.0'
    implementation 'com.google.android.material:material:1.4.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.2'
    testImplementation 'junit:junit:4.+'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
}
```

} Add these two dependencies to build.gradle file

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.jasondataparsingvolley">
<uses-permission android:name="android.permission.INTERNET"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Jasondataparsingvolley">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Add internet permission to manifest file

activity_main.xml

```xml
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingHorizontal="16dp"
        android:orientation="vertical">

        <EditText
            android:id="@+id/user_input"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:hint="Enter user name" />

        <Button
            android:id="@+id/btn_fetch_data"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:text="Fetch Data" />

        <TextView
            android:id="@+id/result_view"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="result" />

        <ImageView
            android:id="@+id/image_view"
            android:layout_width="match_parent"
            android:layout_height="match_parent" />

    </LinearLayout>

</ScrollView>
```

MainActivity.java

```java
package com.example.jasondataparsingvolley;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.toolbox.JsonObjectRequest;
import com.android.volley.toolbox.Volley;
import com.squareup.picasso.Picasso;

import org.json.JSONException;

public class MainActivity extends AppCompatActivity {

    RequestQueue queue;
    EditText userInput;
    Button btnFetchData;
    TextView resultView;
    ImageView imageView;

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

        queue = Volley.newRequestQueue(this);
        userInput = findViewById(R.id.user_input);
        btnFetchData = findViewById(R.id.btn_fetch_data);
        resultView = findViewById(R.id.result_view);
        imageView = findViewById(R.id.image_view);
btnFetchData.setOnClickListener(view -> {
            String url = "https://api.github.com/users/" +userInput.getText();
            JsonObjectRequest request = new JsonObjectRequest(Request.Method.GET,
url, null,
                    response -> {
                        try {
                            String login = response.getString("login");
                            String id = response.getString("id");
                            String nodeId = response.getString("node_id");
                            String avatarUrl = response.getString("avatar_url");
                            resultView.setText("Login: " + login + "\nID: " + id +
"\nNode ID: " + nodeId);
                            Picasso.get().load(avatarUrl).into(imageView);
                        } catch (JSONException e) {
                            e.printStackTrace();
                            Toast.makeText(this, "Something went wrong!",
Toast.LENGTH_SHORT).show();
                        }
                    },
                    error -> {Toast.makeText(this, "User not found!",
Toast.LENGTH_SHORT).show();
    resultView.setText("");
    imageView.setImageDrawable(null);
}

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