CLOUD CONNECTIVITY

Android Glide Image Library – Building Image Gallery App

BY RAVI TAMADA - APRIL 20, 2016 - 347 COMMENTS

SEARCH HERE

Type and hit enter...

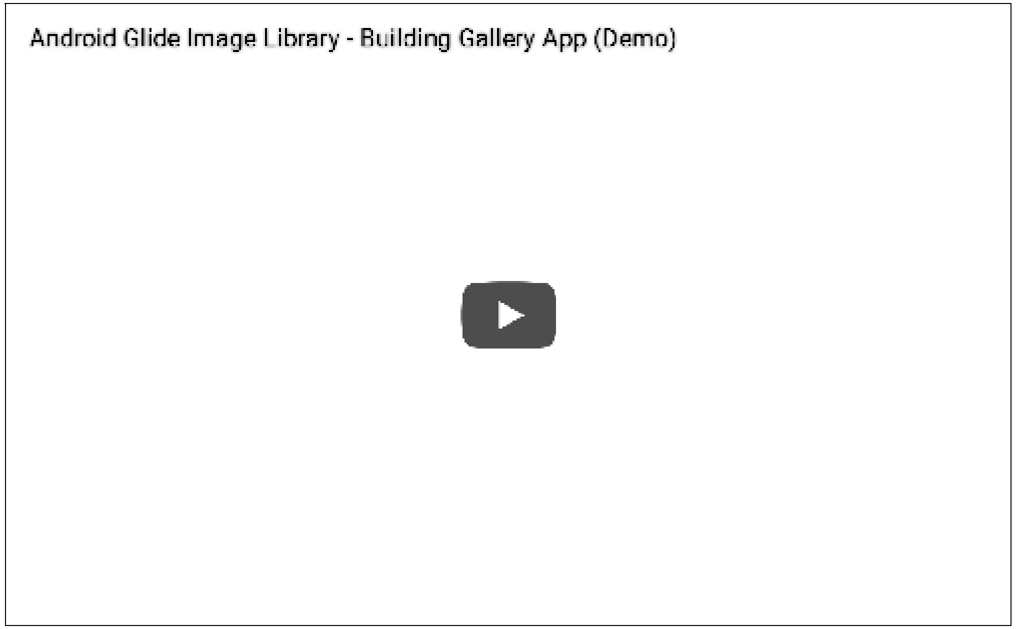
Loading an image from internet is pretty easier using Volley library. But here is a much better solution than volley i.e Glide image library. When compared to volley, Glide wins in lot of scenarios in terms of performance and usability. Below are the advantages of **Glide** over volley

> Supports fetching, decoding, and displaying video stills, images, and animated GIFs
> Placeholder can be added before the loading the media
> Loads thumbnail (blurred) first and then loads the high resolution image like in WhatsApp or Facebook.
> Crossfading effects between the media
> Supports image arbitrary transformations like loading image in circular shape or any other shape.
> Better Memory and disk caching mechanisms
> Works well with both **Volley** and **OkHttp** libraries

This article explains how to build a simple image gallery app where all the images will be loaded from internet. First all the thumbnail images displayed in a grid manner and upon selecting the single image, a fullscreen image slider will be launched.

DOWNLOAD CODE

VIDEO DEMO


Android Glide Image Library - Building Gallery App (Demo)

How to Use It?

Integrating **Glide** in your project is very easy. First add the glide dependency to your **build.gradle**.

```
build.gradle

dependencies {
    // glide
```

CATEGORIES

○ App	(18)
○ Beginner	(44)
○ Cloud Connectivity	(51)
○ Database	(12)
○ Firebase	(8)
○ Material Design	(16)
○ UI & UX	(33)

WE'RE SOCIAL

<div class="fb-page" data-href="https://www.facebook.com/AndroidHive/"

Second load the image into ImageView using below code snippet.

```
String imgUrl = "https://api.androidhive.info/images/glide/medium/deadpool.jpg";

ImageView imageView = (ImageView) view.findViewById(R.id.thumbnail);

Glide.with(mContext).load(imgUrl)
                .thumbnail(0.5f)
                .crossFade()
                .diskCacheStrategy(DiskCacheStrategy.ALL)
                .into(imageView);
```

Sample JSON

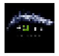
To build the gallery app, I have created a sample JSON which contains the image urls required. Each image is highly **compressed** and resized in three different resolutions i.e **Higher**, **medium** and **smaller**. For the grid display, we load the medium resolution image and for the fullscreen image slider, we load the higher resolution image.

JSON link: https://api.androidhive.info/json/glide.json

```
[{
    "name": "Deadpool",
    "url": {
        "small": "https://api.androidhive.info/images/glide/small/deadpool.jpg",
        "medium": "https://api.androidhive.info/images/glide/medium/deadpool.jpg",
        "large": "https://api.androidhive.info/images/glide/large/deadpool.jpg"
    },
    "timestamp": "February 12, 2016"
},
{
    "name": "Batman vs Superman",
```

facepile='true'><blockquote cite="https://www.facebook.com/AndroidHive/" class="fb-xfbml-parse-ignore"><a href="https://www.facebook.com/AndroidHive/">AndroidHive</a></blockquote></div>

POPULAR ANDROID TUTORIALS


Android SQLite Database Tutorial - 1,749,333 views

How to connect Android with PHP, MySQL - 1,700,591 views

Android JSON Parsing Tutorial - 1,593,082 views

Android Push Notifications using Firebase Cloud Messaging FCM & PHP - 1,521,540 views

Android Sliding Menu using Navigation Drawer - 1,454,335 views

Android Login and Registration with PHP, MySQL and SQLite - 1,291,927 views

Android Custom ListView with Image and Text - 1,131,332 views

Android GPS, Location Manager Tutorial - 961,920 views

Android Tab Layout with Swipeable Views - 839,708 views

```
            "medium": "https://api.androidhive.info/images/glide/medium/bvs.png",
            "large": "https://api.androidhive.info/images/glide/large/bvs.png"
    },
    "timestamp": "March 25, 2016"
}]
```

Now let's start building the image gallery app.

Building Image Gallery App

1. Create a new project in Android Studio from **File** ⇒ **New Project**. When it prompts you to select the default activity, select **Blank Activity** and proceed.

2. Open **build.gradle** and add Glide, Volley and RecyclerView dependencies. **Volley** is used to download the gallery json by making HTTP call. **RecyclerView** is used to show the gallery images in a Grid fashion.

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:23.2.1'
    compile 'com.android.support:design:23.2.1'
    compile 'com.android.support:support-v4:23.2.1'

    // RecyclerView
    compile 'com.android.support:recyclerview-v7:23.1.1'

    // volley
    compile 'com.android.volley:volley:1.0.0'

    // Glide
    compile 'com.github.bumptech.glide:glide:3.7.0'
}
```

4. Create a class named **AppController.java** under **app** package. This is a singleton class in which we initialize the volley's core objects.

```
AppController.java
package info.androidhive.glide.app;

import android.app.Application;
import android.text.TextUtils;

import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.toolbox.Volley;

public class AppController extends Application {

    public static final String TAG = AppController.class
            .getSimpleName();

    private RequestQueue mRequestQueue;

    private static AppController mInstance;

    @Override
    public void onCreate() {
        super.onCreate();
        mInstance = this;
    }

    public static synchronized AppController getInstance() {
        return mInstance;
    }

    public RequestQueue getRequestQueue() {
        if (mRequestQueue == null) {
            mRequestQueue = Volley.newRequestQueue(getApplicationContext());
        }
    }

    public <T> void addToRequestQueue(Request<T> req, String tag) {
        // set the default tag if tag is empty
        req.setTag(TextUtils.isEmpty(tag) ? TAG : tag);
        getRequestQueue().add(req);
    }

    public <T> void addToRequestQueue(Request<T> req) {
        req.setTag(TAG);
        getRequestQueue().add(req);
    }

    public void cancelPendingRequests(Object tag) {
        if (mRequestQueue != null) {
            mRequestQueue.cancelAll(tag);
        }
    }
}
```

5. Open **AndroidManifest.xml** and add the **AppController** to **<application>** tag. Also add the INTERNET permission as we need to make HTTP calls.

```
AndroidManifest.xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="info.androidhive.glide">

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

    <application
        android:name=".app.AppController"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        <activity android:name=".activity.MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

Now our project is ready with all the dependencies added. Let’s start adding the grid gallery first.

Adding the Grid Gallery View

6. Open the layout files of your main activity and add the recyclerView. For my main activity I have two layout files **activity_main.xml** and **content_main.xml**

The **activity_main.xml** contains the general **AppBar** and **Toolbar**.

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout xmlns:android="http://schemas.android.com/apk/res/a
    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:fitsSystemWindows="true"
    tools:context=".activity.MainActivity">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">
```

```
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

    </android.support.design.widget.AppBarLayout>

    <include layout="@layout/content_main" />

</android.support.design.widget.CoordinatorLayout>
```

The **content_main.xml** contains the **recyclerView** to load the images in grid.

```
content_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"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="info.androidhive.glide.activity.MainActivity"
    tools:showIn="@layout/activity_main">

    <android.support.v7.widget.RecyclerView
        android:id="@+id/recycler_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:scrollbars="vertical" />

</RelativeLayout>
```

7. Under **helper** package, create a class named **SquareLayout.java**. This class helps the images to display in square ratio

SquareLayout.java

```java
package info.androidhive.glide.helper;

import android.annotation.TargetApi;
import android.content.Context;
import android.os.Build;
import android.util.AttributeSet;
import android.widget.RelativeLayout;

/**
 * Created by Lincoln on 05/04/16.
 */
class SquareLayout extends RelativeLayout {

    public SquareLayout(Context context) {
        super(context);
    }

    public SquareLayout(Context context, AttributeSet attrs) {
        super(context, attrs);
    }

    public SquareLayout(Context context, AttributeSet attrs, int defStyleAttr) {
        super(context, attrs, defStyleAttr);
    }

    @TargetApi(Build.VERSION_CODES.LOLLIPOP)
    public SquareLayout(Context context, AttributeSet attrs, int defStyleAttr, int defStyleRes) {
        super(context, attrs, defStyleAttr, defStyleRes);
    }

    @Override
    protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
        // Set a square layout.
        super.onMeasure(widthMeasureSpec, widthMeasureSpec);
    }
}
```

gallery_thumbnail.xml

```
<?xml version="1.0" encoding="utf-8"?>
<info.androidhive.glide.helper.SquareLayout xmlns:android="http://schemas.android.com/apk/res/androi
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <ImageView
        android:id="@+id/thumbnail"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:adjustViewBounds="true"
        android:scaleType="centerCrop" />
</info.androidhive.glide.helper.SquareLayout>
```

9. Under **adapter** package, create a class named **GalleryAdapter.java** This is a adapter class which inflates the **gallery_thumbnail.xml** and renders the images in recyclerView.

GalleryAdapter.java

```
package info.androidhive.glide.adapter;

import android.content.Context;
import android.support.v7.widget.RecyclerView;
import android.view.GestureDetector;
import android.view.LayoutInflater;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;

import com.bumptech.glide.Glide;
import com.bumptech.glide.load.engine.DiskCacheStrategy;
import info.androidhive.glide.R;
import info.androidhive.glide.model.Image;

/**
 * Created by Lincoln on 31/03/16.
 */

public class GalleryAdapter extends RecyclerView.Adapter<GalleryAdapter.MyViewHolder> {

    private List<Image> images;
    private Context mContext;

    public class MyViewHolder extends RecyclerView.ViewHolder {
        public ImageView thumbnail;

        public MyViewHolder(View view) {
            super(view);
            thumbnail = (ImageView) view.findViewById(R.id.thumbnail);
        }
    }


    public GalleryAdapter(Context context, List<Image> images) {
        mContext = context;
        this.images = images;
    }

    @Override
    public MyViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        View itemView = LayoutInflater.from(parent.getContext())
                .inflate(R.layout.gallery_thumbnail, parent, false);

        return new MyViewHolder(itemView);
    }

    @Override
    public void onBindViewHolder(MyViewHolder holder, int position) {
        Image image = images.get(position);
```

```
                .crossFade()
                .diskCacheStrategy(DiskCacheStrategy.ALL)
                .into(holder.thumbnail);
    }

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

    public interface ClickListener {
        void onClick(View view, int position);

        void onLongClick(View view, int position);
    }

    public static class RecyclerTouchListener implements RecyclerView.OnItemTouchListener {

        private GestureDetector gestureDetector;
        private GalleryAdapter.ClickListener clickListener;

        public RecyclerTouchListener(Context context, final RecyclerView recyclerView, final Gallery
            this.clickListener = clickListener;
            gestureDetector = new GestureDetector(context, new GestureDetector.SimpleOnGestureLister
                @Override
                public boolean onSingleTapUp(MotionEvent e) {
                    return true;
                }

                @Override
                public void onLongPress(MotionEvent e) {
                    View child = recyclerView.findChildViewUnder(e.getX(), e.getY());
                    if (child != null && clickListener != null) {
                        clickListener.onLongClick(child, recyclerView.getChildPosition(child));
                    }
                }
            });
        }
```

```
            View child = rv.findChildViewUnder(e.getX(), e.getY());
            if (child != null && clickListener != null && gestureDetector.onTouchEvent(e)) {
                clickListener.onClick(child, rv.getChildPosition(child));
            }
            return false;
        }

        @Override
        public void onTouchEvent(RecyclerView rv, MotionEvent e) {
        }

        @Override
        public void onRequestDisallowInterceptTouchEvent(boolean disallowIntercept) {

        }
    }
}
```

10. Finally open **MainActivity.java** and do the below changes

> Download the json by making volley http request. **fetchImages()** method is used for this purpose

> Parse the json and add the models to array list.

MainActivity.java

```
package info.androidhive.glide.activity;

import android.app.ProgressDialog;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.DefaultItemAnimator;
import android.support.v7.widget.GridLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.support.v7.widget.Toolbar;
import android.util.Log;

import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.JsonArrayRequest;

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

import java.util.ArrayList;

import info.androidhive.glide.R;
import info.androidhive.glide.adapter.GalleryAdapter;
import info.androidhive.glide.app.AppController;
import info.androidhive.glide.model.Image;

public class MainActivity extends AppCompatActivity {

    private String TAG = MainActivity.class.getSimpleName();
    private static final String endpoint = "https://api.androidhive.info/json/glide.json";
    private ArrayList<Image> images;
    private ProgressDialog pDialog;
    private GalleryAdapter mAdapter;
    private RecyclerView recyclerView;

    @Override
```

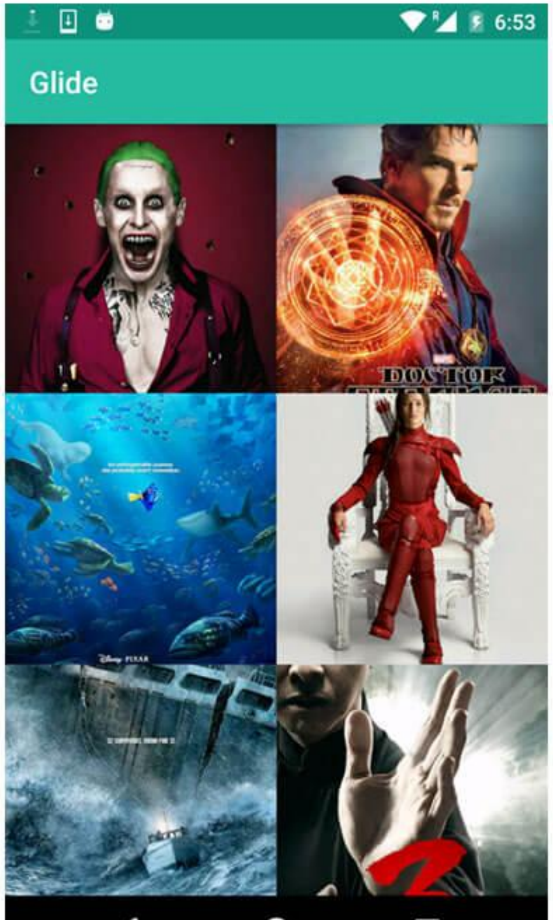
```
        setContentView(R.layout.activity_main);

        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        recyclerView = (RecyclerView) findViewById(R.id.recycler_view);

        pDialog = new ProgressDialog(this);
        images = new ArrayList<>();
        mAdapter = new GalleryAdapter(getApplicationContext(), images);

        RecyclerView.LayoutManager mLayoutManager = new GridLayoutManager(getApplicationContext(), 2
        recyclerView.setLayoutManager(mLayoutManager);
        recyclerView.setItemAnimator(new DefaultItemAnimator());
        recyclerView.setAdapter(mAdapter);

        /* recyclerView.addOnItemTouchListener(new GalleryAdapter.RecyclerTouchListener(getApplicati
            @Override
            public void onClick(View view, int position) {
                Bundle bundle = new Bundle();
                bundle.putSerializable("images", images);
                bundle.putInt("position", position);

                FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
                SlideshowDialogFragment newFragment = SlideshowDialogFragment.newInstance();
                newFragment.setArguments(bundle);
                newFragment.show(ft, "slideshow");
            }

            @Override
            public void onLongClick(View view, int position) {

            }
        }));*/

        fetchImages();
    }

    private void fetchImages() {
```

```
pDialog.show();

JsonArrayRequest req = new JsonArrayRequest(endpoint,
        new Response.Listener<JSONArray>() {
            @Override
            public void onResponse(JSONArray response) {
                Log.d(TAG, response.toString());
                pDialog.hide();

                images.clear();
                for (int i = 0; i < response.length(); i++) {
                    try {
                        JSONObject object = response.getJSONObject(i);
                        Image image = new Image();
                        image.setName(object.getString("name"));

                        JSONObject url = object.getJSONObject("url");
                        image.setSmall(url.getString("small"));
                        image.setMedium(url.getString("medium"));
                        image.setLarge(url.getString("large"));
                        image.setTimestamp(object.getString("timestamp"));

                        images.add(image);

                    } catch (JSONException e) {
                        Log.e(TAG, "Json parsing error: " + e.getMessage());
                    }
                }

                mAdapter.notifyDataSetChanged();
            }
        }, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        Log.e(TAG, "Error: " + error.getMessage());
        pDialog.hide();
    }
});
```

```
    }
}
```

If you run the app, you can see the images displayed in grid manner. Be sure that your device is connected to internet.

Android Image Gallery - using Glide



Fullscreen Image Slideshow

Now we'll see how to build a fullscreen image slider with swiping functionality. We use a **DialogFragment** and **ViewPager** for this purpose.

11. Create a layout named **image_fullscreen_preview.xml** under **res** ⇒ **layout**. This layout is used to display the image in fullscreen view.

image_fullscreen_preview.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/RelativeLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@android:color/black">

    <ImageView
        android:id="@+id/image_preview"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_centerInParent="true"
        android:scaleType="fitCenter" />

</RelativeLayout>
```

12. Under **activity** package, create a class named **SlideshowDialogFragment.java**. This is a fragment class which extends DialogFragment.

```
package info.androidhive.glide.activity;

import android.content.Context;
import android.os.Bundle;
import android.support.v4.app.DialogFragment;
import android.support.v4.view.PagerAdapter;
import android.support.v4.view.ViewPager;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

import com.bumptech.glide.Glide;
import com.bumptech.glide.load.engine.DiskCacheStrategy;

import java.util.ArrayList;

import info.androidhive.glide.R;
import info.androidhive.glide.model.Image;


public class SlideshowDialogFragment extends DialogFragment {
    private String TAG = SlideshowDialogFragment.class.getSimpleName();
    private ArrayList<Image> images;
    private ViewPager viewPager;
    private MyViewPagerAdapter myViewPagerAdapter;
    private TextView lblCount, lblTitle, lblDate;
    private int selectedPosition = 0;

    static SlideshowDialogFragment newInstance() {
        SlideshowDialogFragment f = new SlideshowDialogFragment();
        return f;
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
```

```
        lblCount = (TextView) v.findViewById(R.id.lbl_count);
        lblTitle = (TextView) v.findViewById(R.id.title);
        lblDate = (TextView) v.findViewById(R.id.date);

        images = (ArrayList<Image>) getArguments().getSerializable("images");
        selectedPosition = getArguments().getInt("position");

        Log.e(TAG, "position: " + selectedPosition);
        Log.e(TAG, "images size: " + images.size());

        myViewPagerAdapter = new MyViewPagerAdapter();
        viewPager.setAdapter(myViewPagerAdapter);
        viewPager.addOnPageChangeListener(viewPagerPageChangeListener);

        setCurrentItem(selectedPosition);

        return v;
    }

    private void setCurrentItem(int position) {
        viewPager.setCurrentItem(position, false);
        displayMetaInfo(selectedPosition);
    }

    //  page change listener
    ViewPager.OnPageChangeListener viewPagerPageChangeListener = new ViewPager.OnPageChangeListener(

        @Override
        public void onPageSelected(int position) {
            displayMetaInfo(position);
        }

        @Override
        public void onPageScrolled(int arg0, float arg1, int arg2) {

        }

        @Override
        public void onPageScrollStateChanged(int arg0) {
    };

    private void displayMetaInfo(int position) {
        lblCount.setText((position + 1) + " of " + images.size());

        Image image = images.get(position);
        lblTitle.setText(image.getName());
        lblDate.setText(image.getTimestamp());
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setStyle(DialogFragment.STYLE_NORMAL, android.R.style.Theme_Black_NoTitleBar_Fullscreen);
    }

    //  adapter
    public class MyViewPagerAdapter extends PagerAdapter {

        private LayoutInflater layoutInflater;

        public MyViewPagerAdapter() {
        }

        @Override
        public Object instantiateItem(ViewGroup container, int position) {

            layoutInflater = (LayoutInflater) getActivity().getSystemService(Context.LAYOUT_INFLATER
            View view = layoutInflater.inflate(R.layout.image_fullscreen_preview, container, false);

            ImageView imageViewPreview = (ImageView) view.findViewById(R.id.image_preview);

            Image image = images.get(position);

            Glide.with(getActivity()).load(image.getLarge())
                    .thumbnail(0.5f)
                    .crossFade()
                    .diskCacheStrategy(DiskCacheStrategy.ALL)
                    .into(imageViewPreview);
```

```
            return view;
        }

        @Override
        public int getCount() {
            return images.size();
        }

        @Override
        public boolean isViewFromObject(View view, Object obj) {
            return view == ((View) obj);
        }


        @Override
        public void destroyItem(ViewGroup container, int position, Object object) {
            container.removeView((View) object);
        }
    }
}
```

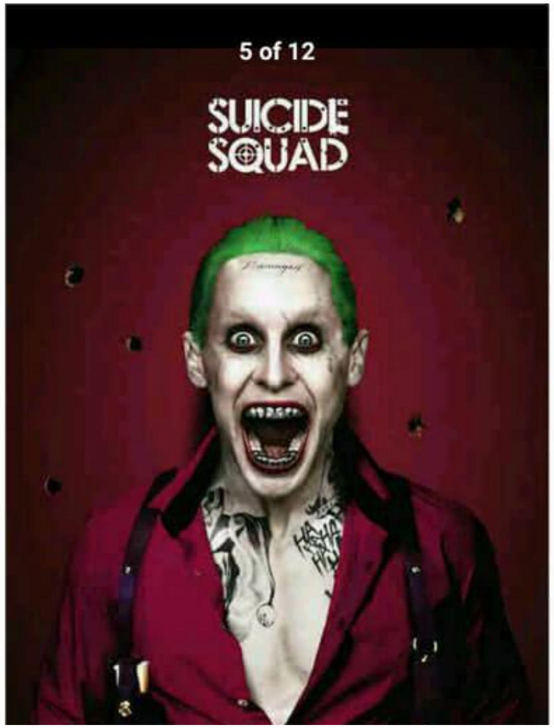
13. Open **MainActivity.java** and add the click event to **recyclerView** in **onCreate()** method. (This code is already provided in above step, just uncomment it)

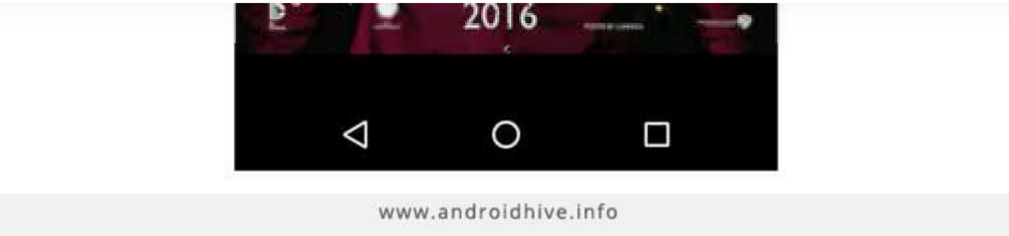
MainActivity.java

```
recyclerView.addOnItemTouchListener(new GalleryAdapter.RecyclerTouchListener(getApplicationContext()
    @Override
    public void onClick(View view, int position) {
        Bundle bundle = new Bundle();
        bundle.putSerializable("images", images);
        bundle.putInt("position", position);

        FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
        SlideshowDialogFragment newFragment = SlideshowDialogFragment.newInstance();
    }

    @Override
    public void onLongClick(View view, int position) {

    }
}));
```

Run the app once more and try tapping on thumbnail image. You should see the fullscreen image slider with swiping functionality enabled.

Android Fullscreen Image Slider - using Glide





Ravi Tamada

Ravi is hardcore Android programmer and Android programming has been his passion since he compiled his first hello-world program. Solving real problems of Android developers through tutorials has always been interesting part for him.

GLIDE GRID JSON RECYCLERVIEW SWIPE VIEW PAGER VOLLEY

RELATED POSTS