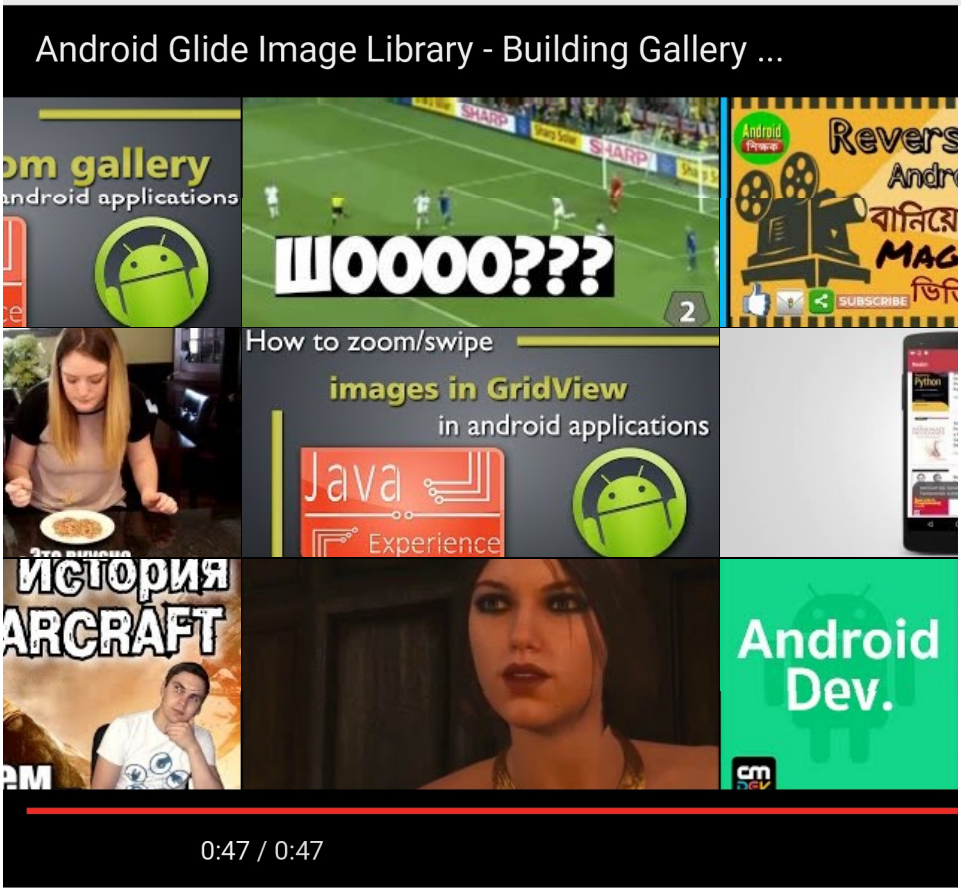
Android Glide Image Library – Building Image Gallery App

by **Ravi Tamada** / April 20, 2016 / **94 Comments**

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



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
```

1. Android SQLite Database Tutorial - 1,400,097 views
2. How to connect Android with PHP, MySQL - 1,379,965 views
3. Android JSON Parsing Tutorial - 1,232,394 views
4. Android Push Notifications using Google Cloud Messaging (GCM), PHP and MySQL - 1,165,292 views
5. Android Sliding Menu using Navigation Drawer - 1,064,672 views
6. Android Custom ListView with Image and Text - 963,501 views
7. Android Login and

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

Second load the image into ImageView using below code snippet.

```
String imgUrl = "http://api.androidhive.info/images/g

ImageView imageView = (ImageView) view.findViewById(R

Glide.with(mContext).load(imgUrl)
                .thumbnail(0.5f)
                .crossFade()
                .diskCacheStrategy(DiskCacheStrategy.
                .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: http://api.androidhive.info/json/glide.json

```
[{
    "name": "Deadpool",
    "url": {
        "small": "http://api.androidhive.info/images/
        "medium": "http://api.androidhive.info/images
        "large": "http://api.androidhive.info/images/
    },
    "timestamp": "February 12, 2016"
},
{
    "name": "Batman vs Superman",
    "url": {
        "small": "http://api.androidhive.info/images/
        "medium": "http://api.androidhive.info/images
        "large": "http://api.androidhive.info/images/
    },
    "timestamp": "March 25, 2016"
}]
```

Registration with PHP, MySQL and SQLite - 934,746 views

8. Android GPS, Location Manager Tutorial - 710,472 views

9. Android Tab Layout with Swipeable Views - 673,150 views

10. Android working with Google Maps V2 - 596,910 views

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

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

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

3. Create three packages named **activity**, **adapter**, **app**, **model** and **helper** and place your **MainActivity.java** under **activity** package. These packages helps in keeping your project organized.

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.cl
            .getSimpleName();

    private RequestQueue mRequestQueue;

    private static AppController mInstance;

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

    public static synchronized AppController getInsta
        return mInstance;
    }

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

        return mRequestQueue;
    }

    public <T> void addToRequestQueue(Request<T> req,
        // 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/a
    package="info.androidhive.glide">

    <uses-permission android:name="android.permission

    <application
        android:name=".app.AppController"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".activity.MainActivit
            <intent-filter>
                <action android:name="android.intent.

                <category android:name="android.inten
            </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 xmln
    xmlns:app="http://schemas.android.com/apk/res-aut
    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.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSiz
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOver

    </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
    xmlns:app="http://schemas.android.com/apk/res-aut
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_vie
    tools:context="info.androidhive.glide.activity.Ma
    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 in grid view.

SquareLayout.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
        super(context, attrs);
    }

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

    @TargetApi(Build.VERSION_CODES.LOLLIPOP)
    public SquareLayout(Context context, AttributeSet
        super(context, attrs, defStyleAttr, defStyleR
    }

    @Override
    protected void onMeasure(int widthMeasureSpec, in
        // Set a square layout.
        super.onMeasure(widthMeasureSpec, widthMeasur
    }
}
```

8. Under **res ⇒ layout**, create a layout named **gallery_thumbnail.xml**. This layout contains an **ImageView** to display the thumbnail image in gallery view.

gallery_thumbnail.xml

```
<?xml version="1.0" encoding="utf-8"?>
<info.androidhive.glide.helper.SquareLayout xmlns:andr
    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.DiskCacheStrate

import java.util.List;

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

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

public class GalleryAdapter extends RecyclerView.Adap

    private List<Image> images;
    private Context mContext;

    public class MyViewHolder extends RecyclerView.Vi
        public ImageView thumbnail;

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

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

@Override
public MyViewHolder onCreateViewHolder(ViewGroup
    View itemView = LayoutInflater.from(parent.ge
            .inflate(R.layout.gallery_thumbnail,

    return new MyViewHolder(itemView);
}

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

    Glide.with(mContext).load(image.getMedium())
            .thumbnail(0.5f)
            .crossFade()
            .diskCacheStrategy(DiskCacheStrategy.
            .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 impleme

    private GestureDetector gestureDetector;
    private GalleryAdapter.ClickListener clickLis

    public RecyclerTouchListener(Context context,
        this.clickListener = clickListener;
        gestureDetector = new GestureDetector(con
            @Override
            public boolean onSingleTapUp(MotionEv
                return true;
            }

            @Override
            public void onLongPress(MotionEvent e
                View child = recyclerView.findChi
                if (child != null && clickListene
                    clickListener.onLongClick(chi
                }
            }
        });
```

```
        }

        @Override
        public boolean onInterceptTouchEvent(RecyclerV

            View child = rv.findChildViewUnder(e.getX
            if (child != null && clickListener != nul
                clickListener.onClick(child, rv.getCh
            }
            return false;
        }

        @Override
        public void onTouchEvent(RecyclerView rv, Mot
        }

        @Override
        public void onRequestDisallowInterceptTouchEv

        }
    }
}
```

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.

> Pass the array list to recyclerView's adapter class.

MainActivity.java

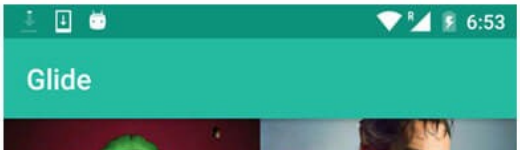
```
package info.androidhive.glide.activity;

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.getSimpleN
    private static final String endpoint = "http://ap
    private ArrayList<Image> images;
    private ProgressDialog pDialog;
    private GalleryAdapter mAdapter;
    private RecyclerView recyclerView;

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

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

        recyclerView = (RecyclerView) findViewById(R.

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

        RecyclerView.LayoutManager mLayoutManager = n
        recyclerView.setLayoutManager(mLayoutManager)
        recyclerView.setItemAnimator(new DefaultItemA
        recyclerView.setAdapter(mAdapter);

        /* recyclerView.addOnItemTouchListener(new Ga
            @Override
            public void onClick(View view, int positi
                Bundle bundle = new Bundle();
                bundle.putSerializable("images", imag
                bundle.putInt("position", position);

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

            @Override
            public void onLongClick(View view, int po

            }
        }));*/

        fetchImages();
    }

    private void fetchImages() {
```

```
        pDialog.setMessage("Downloading json...");
        pDialog.show();

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

                        images.clear();
                        for (int i = 0; i < response.
                            try {
                                JSONObject object = r
                                Image image = new Ima
                                image.setName(object.

                                JSONObject url = obje
                                image.setSmall(url.ge
                                image.setMedium(url.g
                                image.setLarge(url.ge
                                image.setTimestamp(ob

                                images.add(image);

                            } catch (JSONException e)
                                Log.e(TAG, "Json pars
                            }
                        }

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

        // Adding request to request queue
        AppController.getInstance().addToRequestQueue
    }
}
```

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



www.androidhive.info

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
    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.

DialogFragment.java

```
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.DiskCacheStrate

import java.util.ArrayList;

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


public class SlideshowDialogFragment extends DialogFr
    private String TAG = SlideshowDialogFragment.class
    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 SlideshowDial
        return f;
    }

    @Override
    public View onCreateView(LayoutInflater inflater,
                             Bundle savedInstanceStat
        View v = inflater.inflate(R.layout.fragment_i
        viewPager = (ViewPager) v.findViewById(R.id.v
        lblCount = (TextView) v.findViewById(R.id.lbl
        lblTitle = (TextView) v.findViewById(R.id.tit
        lblDate = (TextView) v.findViewById(R.id.date

        images = (ArrayList<Image>) getArguments().ge
        selectedPosition = getArguments().getInt("pos

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

```
        myViewPagerAdapter = new MyViewPagerAdapter()
        viewPager.setAdapter(myViewPagerAdapter);
        viewPager.addOnPageChangeListener(viewPagerPa

        setCurrentItem(selectedPosition);

        return v;
    }

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

    //  page change listener
    ViewPager.OnPageChangeListener viewPagerPageChang

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

        @Override
        public void onPageScrolled(int arg0, float ar

        }

        @Override
        public void onPageScrollStateChanged(int arg0

        }
    };

    private void displayMetaInfo(int position) {
        lblCount.setText((position + 1) + " of " + im

        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
    }

    //  adapter
    public class MyViewPagerAdapter extends PagerAdap

        private LayoutInflater layoutInflater;

        public MyViewPagerAdapter() {
        }

        @Override
        public Object instantiateItem(ViewGroup conta
```

```
            layoutInflater = (LayoutInflater) getActi
            View view = layoutInflater.inflate(R.layo

            ImageView imageViewPreview = (ImageView)

            Image image = images.get(position);

            Glide.with(getActivity()).load(image.getL
                    .thumbnail(0.5f)
                    .crossFade()
                    .diskCacheStrategy(DiskCacheStrat
                    .into(imageViewPreview);

            container.addView(view);

            return view;
        }

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

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


        @Override
        public void destroyItem(ViewGroup container,
            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)

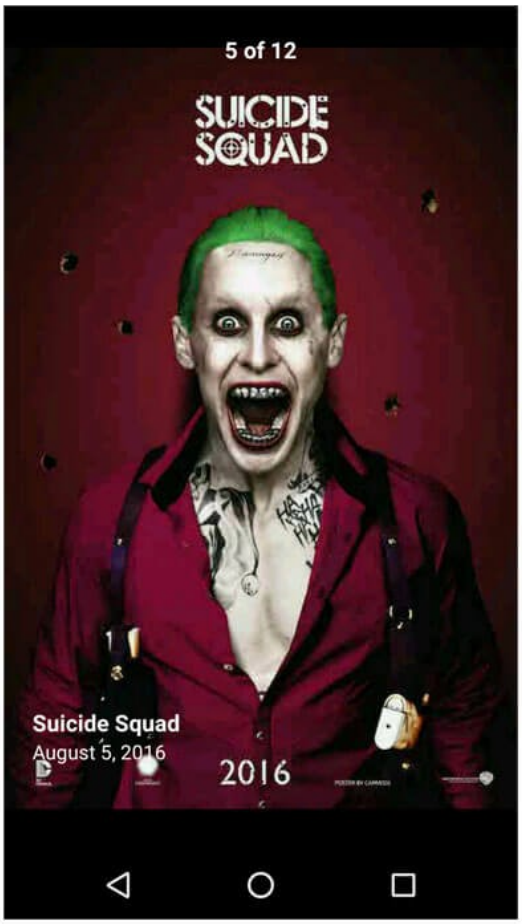
MainActivity.java

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

        FragmentTransaction ft = getSupportFragmentMa
        SlideshowDialogFragment newFragment = Slidesh
        newFragment.setArguments(bundle);
        newFragment.show(ft, "slideshow");
```

```
    }

    @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



www.androidhive.info

ABOUT THE AUTHOR

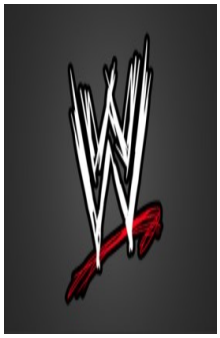
Ravi Tamada

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.

RELATED POSTS

Android Multilevel ListView Tutorial

by **Ravi Tamada**

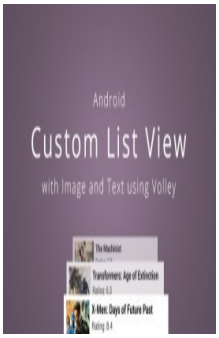


How to implement Android Splash Screen

by **Ravi Tamada**

Android Building Group Chat App using Sockets – Part 2

by **Ravi Tamada**



Android Custom ListView with Image and Text using Volley

by **Ravi Tamada**

Comments Community 1 Login

Recommend 7 Share Sort by Newest

Join the discussion…

Tatenda Kabike • a day ago

Great tutorial Ravi, i implemented this tutorial by replacing Glide with fresco library from facebook the performance still good, i would want to know which image library is better between Glide and Fresco,

Reply • Share ›

mohd khalid Siddiqui • 3 days ago

fullscreen image slider with swiping functionality not working

Reply • Share ›

sam jacky • 4 days ago

is there anyone who have download source code if so plz mail me at samjacky80@gmail.com

Reply • Share ›

Eswar • 4 days ago

Can it be used to show images from phone ? I mean I need to show all images and videos from sd card. Can someone guide me

Reply • Share ›

Ravi Tamada Mod ➚ Eswar • 4 days ago

Yes, everything is same. Just give the path of SD Card image. It should be like file:///_your_path_to_image

Reply • Share ›

Eswar ➚ Ravi Tamada • 4 days ago

In which file? and I need to show all images from SD card. and how can i get the image paths of all images. Thanks in advance.

Reply • Share ›

tamer • 12 days ago

Hello ravi can i integrate a video with this library ? kindly advice .

Reply • Share ›

Ravi Tamada Mod ➚ tamer • 11 days ago

Glide supports Gifs and Videos.

https://futurestud.io/blog/gli