# Android Upload Image using Android Upload Service

June 10, 2016 by Belal Khan — 79 Comments

Hello guys, in this post I came up with an easy solution for uploading files from android to server. So today we will see an example of **Android Upload Image to Server**. I have already posted some example of **Android Upload Image** to Server previously. But in this post we will use **android upload service** for our Android Upload Image App. You can also check the previous tutorials I posted about uploading images from android to server from the given links.

- Android Volley Tutorial to Upload Image to Server
- Android Upload Image From Gallery With Text
- Android Upload Image to Server Using PHP MySQL

In this tutorial I will store the image file inside servers directory and in database I will store the URL of the image. For this I will use Android Upload Service library and it makes uploading files super easy. So lets see how we can do it. First we will create our server side codes.

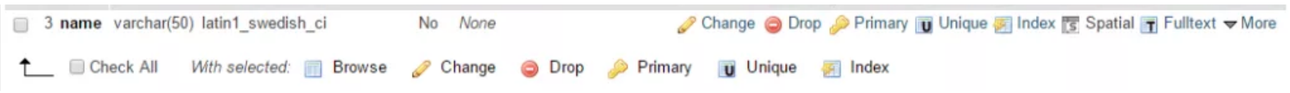
## Android Upload Image to Server Video Tutorial

- You can also go through this video tutorial to learn how to upload image from android to server.

Android Upload Image to Server Tutorial - Creating Server Side S...

## Creating Server Side Codes for Android Upload Image

The first thing we need is to create our server side web services. For server side I am using PHP and MySQL. And for this I am using Wamp server. You can still use xampp or any other application. Now follow the steps to create your web service to handle the file upload.


3 name varchar(50) latin1_swedish_ci No None
Change Drop Primary Unique Index Spatial Fulltext More
Check All With selected: Browse Change Drop Primary Unique Index

- Now inside your server's root directory **(c:/wamp/www)** and create a new folder. I created **AndroidUploadImage**.
- Inside the folder create a folder named **uploads,** in this folder we will save all the uploaded images.
- Create a file named **dbDetails.php** and write the following code.

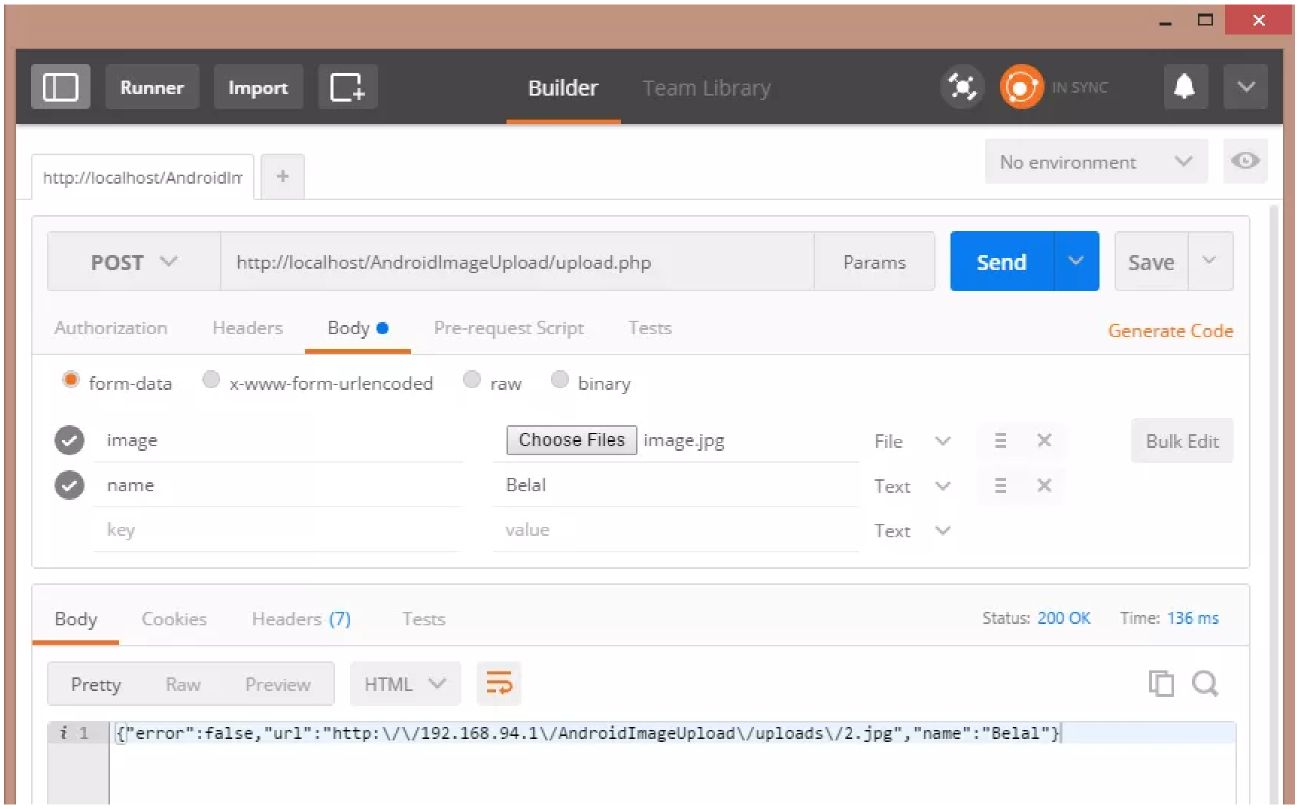
dbDetails.php PHP

```php
<?php
 define('HOST','localhost');
 define('USER','root');
 define('PASS','');
 define('DB','db_images');
```

- Now create a file named **upload.php** and write the following code.

upload.php PHP

```php
<?php

//importing dbDetails file
require_once 'dbDetails.php';

//this is our upload folder
$upload_path = 'uploads/';

//Getting the server ip
$server_ip = gethostbyname(gethostname());

//creating the upload url
$upload_url = 'http://'.$server_ip.'/AndroidImageUpload/'.$upload_path;

//response array
$response = array();


if($_SERVER['REQUEST_METHOD']=='POST'){

//checking the required parameters from the request
if(isset($_POST['name']) and isset($_FILES['image']['name'])){

//connecting to the database
$con = mysqli_connect(HOST,USER,PASS,DB) or die('Unable to Connect...');

//getting name from the request
$name = $_POST['name'];

//file url to store in the database
$file_url = $upload_url . getFileName() . '.' . $extension;

//file path to upload in the server
$file_path = $upload_path . getFileName() . '.'. $extension;

//trying to save the file in the directory
try{
//saving the file
move_uploaded_file($_FILES['image']['tmp_name'],$file_path);
$sql = "INSERT INTO `db_images`.`images` (`id`, `url`, `name`) VALUES (NULL, '$file_url', '$name')

//adding the path and name to database
if(mysqli_query($con,$sql)){

//filling response array with values
$response['error'] = false;
$response['url'] = $file_url;
$response['name'] = $name;
}
//if some error occurred
}catch(Exception $e){
$response['error']=true;
$response['message']=$e->getMessage();
}
//displaying the response
echo json_encode($response);

//closing the connection
mysqli_close($con);
}else{
$response['error']=true;
$response['message']='Please choose a file';
}
}

/*
We are generating the file name
so this method will return a file name for the image to be upload
*/
function getFileName(){
$con = mysqli_connect(HOST,USER,PASS,DB) or die('Unable to Connect...');
$sql = "SELECT max(id) as id FROM images";
$result = mysqli_fetch_array(mysqli_query($con,$sql));

mysqli_close($con);
if($result['id']==null)
return 1;
else
```



POST http://localhost/AndroidImageUpload/upload.php

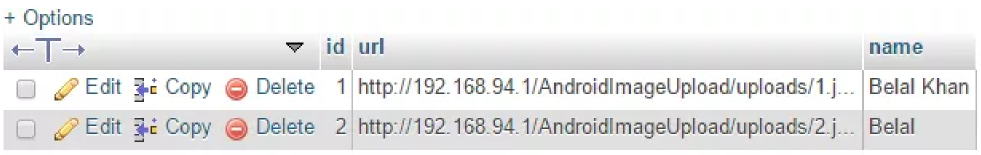
Body: form-data

| key | value | type |
|---|---|---|
| image | image.jpg | File |
| name | Belal | Text |

Status: 200 OK Time: 136 ms

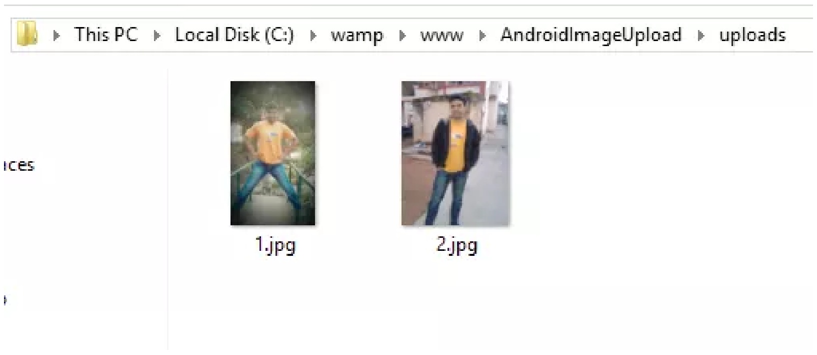
```
{"error":false,"url":"http:\/\/192.168.94.1\/AndroidImageUpload\/uploads\/2.jpg","name":"Belal"}
```

- If you are seeing the above response then. Your script is working fine. You can check the database and upload folder which you have created.



| | id | url | name |
|---|---|---|---|
| Edit Copy Delete | 1 | http://192.168.94.1/AndroidImageUpload/uploads/1.j... | Belal Khan |
| Edit Copy Delete | 2 | http://192.168.94.1/AndroidImageUpload/uploads/2.j... | Belal |

**Database**



This PC ▸ Local Disk (C:) ▸ wamp ▸ www ▸ AndroidImageUpload ▸ uploads

**Upload Directory**

- So its working absolutely fine. Now lets move ahead and create a android project.

- Create a Android Studio Project.
- Create a class named **Constants.java** and write the following code. The following class contains the path to our php file which we created.  You are seeing two strings. The second it the path to the file we will create at the end of this post.

Constants.java Java

```java
package net.simplifiedcoding.androidimageupload;

/**
 * Created by Belal on 6/10/2016.
 */
public class Constants {
    public static final String UPLOAD_URL = "http://192.168.94.1/AndroidImageUpload/upload.php";
    public static final String IMAGES_URL = "http://192.168.94.1/AndroidImageUpload/getImages.php";
}
```

- You need to change the IP according to your system. To know the IP you can use **IPCONFIG** command in command prompt (windows user).
- Now we need to add **android upload service** to our project.

## Adding Android Upload Service

- Go to your app level **build.gradle** file and add the following line inside dependencies block and sync your project.

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:23.4.0'

    //Add this line
    compile 'net.gotev:uploadservice:2.1'
}
```

- Come to **activity_main.xml** and write the following xml code.

activity_main.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
```

```
        android:gravity="center_horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <Button
            android:id="@+id/buttonChoose"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Select" />

        <EditText
            android:id="@+id/editTextName"
            android:hint="Name For Image"
            android:layout_weight="1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

        <Button
            android:id="@+id/buttonUpload"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Upload" />

    </LinearLayout>

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


</LinearLayout>
```

- The above code will generate the following layout.

- As you can see we have two buttons, one to select image and other to upload image. We also have an EditText to enter the name for the image.
- Now come to **MainActivity.java** and write the following code.

MainActivity.java Java

```java
package net.simplifiedcoding.androidimageupload;

import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.graphics.Bitmap;
import android.net.Uri;
import android.os.Bundle;
import android.provider.MediaStore;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.Toast;

import net.gotev.uploadservice.MultipartUploadRequest;
import net.gotev.uploadservice.UploadNotificationConfig;

import java.io.IOException;
import java.util.UUID;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {

    //Image request code
    private int PICK_IMAGE_REQUEST = 1;

    //storage permission code
    private static final int STORAGE_PERMISSION_CODE = 123;

    //Bitmap to get image from gallery
    private Bitmap bitmap;

    //Uri to store the image uri
    private Uri filePath;

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

        //Requesting storage permission
        requestStoragePermission();

        //Initializing views
        buttonChoose = (Button) findViewById(R.id.buttonChoose);
        buttonUpload = (Button) findViewById(R.id.buttonUpload);
        imageView = (ImageView) findViewById(R.id.imageView);
        editText = (EditText) findViewById(R.id.editTextName);

        //Setting clicklistener
        buttonChoose.setOnClickListener(this);
        buttonUpload.setOnClickListener(this);
    }


    /*
    * This is the method responsible for image upload
    * We need the full image path and the name for the image in this method
    * */
    public void uploadMultipart() {
        //getting name for the image
        String name = editText.getText().toString().trim();

        //getting the actual path of the image
        String path = getPath(filePath);

        //Uploading code
        try {
            String uploadId = UUID.randomUUID().toString();

            //Creating a multi part request

        } catch (Exception exc) {
            Toast.makeText(this, exc.getMessage(), Toast.LENGTH_SHORT).show();
        }
    }


    //method to show file chooser
    private void showFileChooser() {
        Intent intent = new Intent();
        intent.setType("image/*");
        intent.setAction(Intent.ACTION_GET_CONTENT);
        startActivityForResult(Intent.createChooser(intent, "Select Picture"), PICK_IMAGE_REQUEST)
    }

    //handling the image chooser activity result
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == PICK_IMAGE_REQUEST && resultCode == RESULT_OK && data != null && data.g
            filePath = data.getData();
            try {
                bitmap = MediaStore.Images.Media.getBitmap(getContentResolver(), filePath);
                imageView.setImageBitmap(bitmap);

            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    //method to get the file path from uri
    public String getPath(Uri uri) {
        Cursor cursor = getContentResolver().query(uri, null, null, null, null);
        cursor.moveToFirst();
        String document_id = cursor.getString(0);
        document_id = document_id.substring(document_id.lastIndexOf(":") + 1);
        cursor.close();

        cursor = getContentResolver().query(
                android.provider.MediaStore.Images.Media.EXTERNAL_CONTENT_URI,
                null, MediaStore.Images.Media._ID + " = ? ", new String[]{document_id}, null);
        cursor.moveToFirst();
        String path = cursor.getString(cursor.getColumnIndex(MediaStore.Images.Media.DATA));
        cursor.close();

        return path;
    }
```

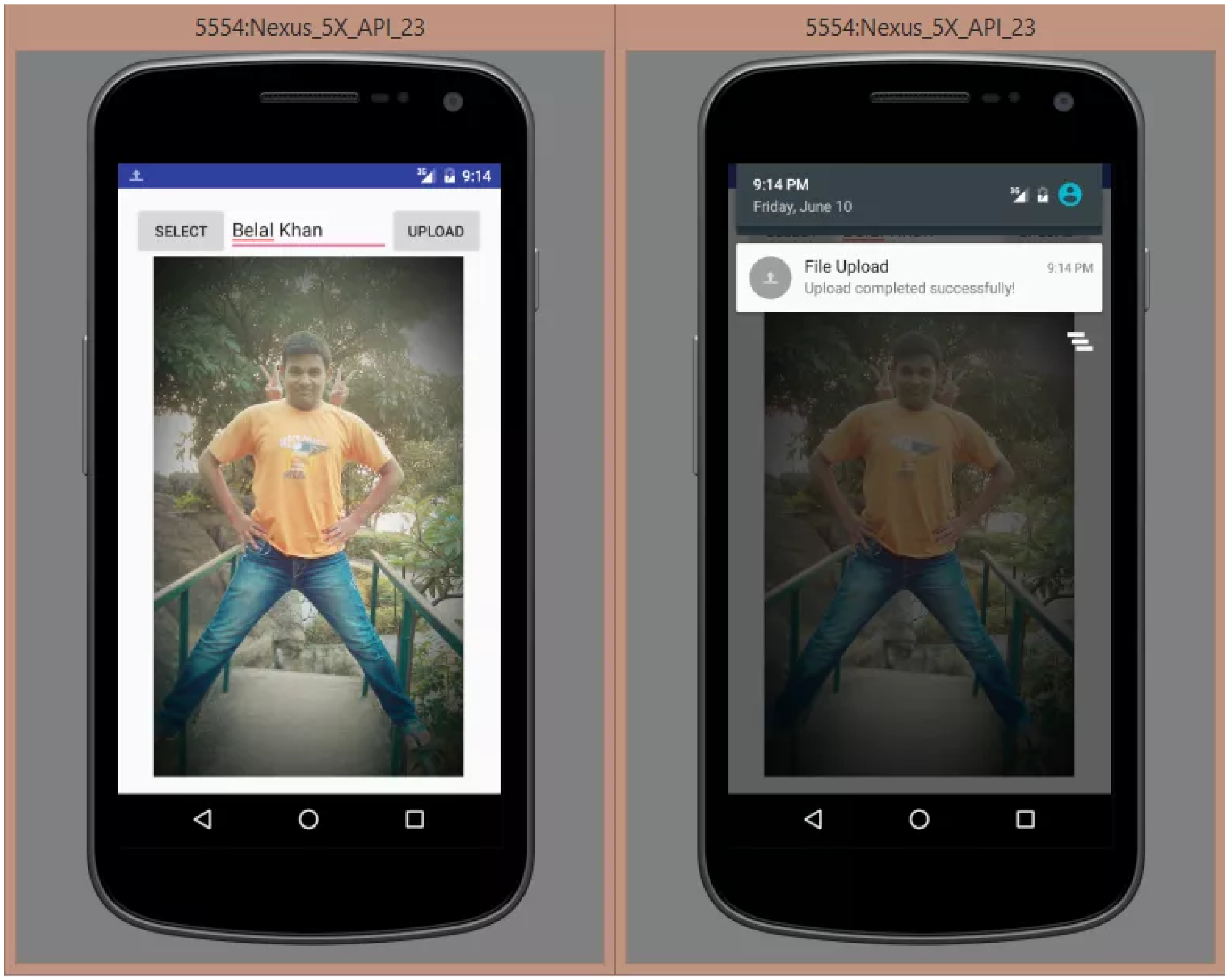
```

        if (ActivityCompat.shouldShowRequestPermissionRationale(this, Manifest.permission.READ_EXTI
            //If the user has denied the permission previously your code will come to this block
            //Here you can explain why you need this permission
            //Explain here why you need this permission
        }
        //And finally ask for the permission
        ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.READ_EXTERNAL_STOI
    }


    //This method will be called when the user will tap on allow or deny
    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNul

        //Checking the request code of our request
        if (requestCode == STORAGE_PERMISSION_CODE) {

            //If permission is granted
            if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                //Displaying a toast
                Toast.makeText(this, "Permission granted now you can read the storage", Toast.LENG
            } else {
                //Displaying another toast if permission is not granted
                Toast.makeText(this, "Oops you just denied the permission", Toast.LENGTH_LONG).show
            }
        }
    }


    @Override
    public void onClick(View v) {
        if (v == buttonChoose) {
            showFileChooser();
        }
        if (v == buttonUpload) {
            uploadMultipart();
        }
    }


}
```

- Finally add the storage and internet permission on your manifest.

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

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

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

</manifest>
```

- Thats it now just run your app.



write the following code.

getImages.php PHP

```php
<?php

 //Importing dbdetails file
 require_once 'dbDetails.php';

 //connection to database
 $con = mysqli_connect(HOST,USER,PASS,DB) or die('Unable to Connect...');

 //sql query to fetch all images
 $sql = "SELECT * FROM images";

 //getting images
 $result = mysqli_query($con,$sql);

 //response array
 $response = array();
 $response['error'] = false;
 $response['images'] = array();

 //traversing through all the rows
 while($row = mysqli_fetch_array($result)){
 $temp = array();
 $temp['id']=$row['id'];
 $temp['name']=$row['name'];
 $temp['url']=$row['url'];
 array_push($response['images'],$temp);
 }
 //displaying the response
 echo json_encode($response);
```

- This code will give you the following JSON.

```json
{
   "error":false,
   "images":[
      {
         "id":"1",
         "name":"Belal Khan",
         "url":"http:\/\/192.168.94.1\/AndroidImageUpload\/uploads\/1.jpg"
      },
      {
         "id":"2",
         "name":"Belal",
         "url":"http:\/\/192.168.94.1\/AndroidImageUpload\/uploads\/2.jpg"
```

```
        {
            "id":"4",
            "name":"Belal Khan",
            "url":"http:\/\/192.168.94.1\/AndroidImageUpload\/uploads\/4.jpg"
        }
    ]
}
```

- Now you can use the following tutorial to display the images using the above JSON.

**Android Custom GridView with Images and Texts using Volley**

- You can also get my source code from the link given below.

**Download is Locked**

Please support us, use one of the buttons below to unlock the content.

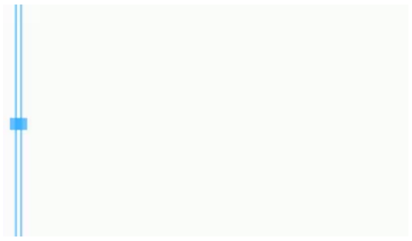
like +1 us tweet

So thats it for this **Android Upload Image** tutorial friends. You can also use this method to upload video and any type of file to your server. Leave your comments if having any doubts or feedbacks. Thank You ☺

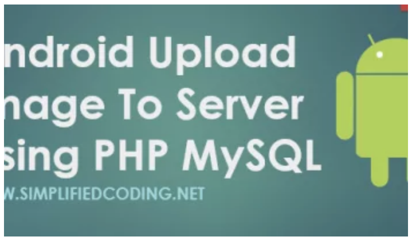
**Sharing is Caring:**

Share 179 Share Tweet Share 1 2 1 point More

**Related**



Android Upload Image to Server Using PHP MySQL
August 26, 2015
In "Android Advance"



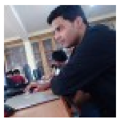
Android Upload Image Using PHP MySQL and Display Images in ListView
September 20, 2015
In "Android Advance"

Android Upload Video to Server using PHP
November 23, 2015
In "Android Advance"

## Some more tutorials for you

6. **Retrofit Upload File Tutorial – Uploading and Downloading Images**

Filed Under: Android Advance, Android Application Development
Tagged With: android upload image, android upload image from gallery, android upload image to server



### About Belal Khan

I am Belal Khan, I am currently pursuing my MCA. In this blog I write tutorials and articles related to coding, app development, android etc.

## Comments

Mahmudur Rahman says
June 11, 2016 at 4:51 am

Assala-mualikum, This is a best tutorial for upload image(base64) to server using volley library. Its done well. Now I want to upload pdf to use base64——– I want know is that possible to upload pdf base64?? ………… If possible I request you to upload a tutorial for pdf upload. Thank you for your great tutorial. Happy Ramadan…