

Android - PHP/MYSQL

In this chapter , we are going to explain, how you can integrate PHP and MYSQL with your android application. This is very useful in case you have a webserver, and you want to access its data on your android application.

MYSQL is used as a database at the webserver and PHP is used to fetch data from the database. Our application will communicate with the PHP page with necessary parameters and PHP will contact MYSQL database and will fetch the result and return the results to us.

PHP - MYSQL

Creating Database

MYSQL database can be created easily using this simple script. The **CREATE DATABASE** statement creates the database.

```
<?php
$con=mysqli_connect("example.com", "username", "password");
$sql ="CREATE DATABASE my_db";
if (mysqli_query($con, $sql)) {
    echo "Database my_db created successfully";
}
?>
```

Creating Tables

Once database is created, its time to create some tables in the database. The **CREATE TABLE** statement creates the database.

```
<?php
$con=mysqli_connect("example.com", "username", "password", "my_db");
$sql ="CREATE TABLE table1 (Username CHAR(30), Password CHAR(30), Role CHAR(30))";
```

```
if (mysqli_query($con,$sql)) {  
    echo "Table have been created successfully";  
}  
?>
```

Inserting Values in tables

When the database and tables are created. Now its time to insert some data into the tables. The **Insert Into** statement creates the database.

```
<?php  
$con=mysqli_connect("example.com","username","password","my_db");  
$sql="INSERT INTO table1 (FirstName, LastName, Age) VALUES ('admin',  
'admin','administrator')";  
if (mysqli_query($con,$sql)) {  
    echo "Values have been inserted successfully";  
}  
?>
```

PHP - GET and POST methods

PHP is also used to fetch the record from the mysql database once it is created. In order to fetch record some information must be passed to PHP page regarding what record to be fetched.

The first method to pass information is through GET method in which **\$_GET** command is used. The variables are passed in the url and the record is fetched. Its syntax is given below –

```
<?php  
$con=mysqli_connect("example.com","username","password","database name");  
  
if (mysqli_connect_errno($con)) {  
    echo "Failed to connect to MySQL: " . mysqli_connect_error();  
}
```

```
$username = $_GET['username'];
$password = $_GET['password'];

$result = mysqli_query($con, "SELECT Role FROM table1 where
Username=' $username'
and Password=' $password'");

$row = mysqli_fetch_array($result);
$data = $row[0];

if($data){
    echo $data;
}

mysqli_close($con);
?>
```

The second method is to use POST method. The only change in the above script is to replace `$_GET` with **`$_POST`**. In Post method, the variables are not passed through URL.

Android - Connecting MYSQL

Connecting Via Get Method

There are two ways to connect to MYSQL via PHP page. The first one is called **Get method**. We will use **HttpGet** and **HttpClient** class to connect. Their syntax is given below –

```
URL url = new URL(link);
HttpClient client = new DefaultHttpClient();
HttpGet request = new HttpGet();
request.setURI(new URI(link));
```

After that you need to call **execute** method of **HttpClient** class and receive it in a **HttpResponse** object. After that you need to open streams to receive the data.

```
HttpResponse response = client.execute(request);  
BufferedReader in = new BufferedReader  
(new InputStreamReader(response.getEntity().getContent()));
```

Connecting Via Post Method

In the Post method, the **URLEncoder**, **URLConnection** class will be used. The **urlencoder** will encode the information of the passing variables. It's syntax is given below –

```
URL url = new URL(link);  
String data = URLEncoder.encode("username", "UTF-8")  
+ "=" + URLEncoder.encode(username, "UTF-8");  
data += "&" + URLEncoder.encode("password", "UTF-8")  
+ "=" + URLEncoder.encode(password, "UTF-8");  
URLConnection conn = url.openConnection();
```

The last thing you need to do is to write this data to the link. After writing, you need to open stream to receive the responded data.

```
OutputStreamWriter wr = new OutputStreamWriter(conn.getOutputStream());  
wr.write( data );  
BufferedReader reader = new BufferedReader(new  
InputStreamReader(conn.getInputStream()));
```

Example

The below example is a complete example of connecting your android application with MySQL database via PHP page. It creates a basic application that allows you to login using GET and POST method.

PHP - MYSQL part

In this example a database with the name of temp has been created at 000webhost.com. In that database, a table has been created with the name of table1. This table has three fields. (Username, Password, Role). The table has only one record which is ("admin","admin","administrator").

The php page has been given below which takes parameters by post method.

```
<?php

$con=mysqli_connect("mysql 10.000webhost.com", "username", "password", "db_name")
;

if (mysqli_connect_errno($con)) {
    echo "Failed to connect to MySQL: " . mysqli_connect_error();
}

$username = $_POST['username'];
$password = $_POST['password'];
$result = mysqli_query($con, "SELECT Role FROM table1 where
Username=' $username' and Password=' $password' ");
$row = mysqli_fetch_array($result);
$data = $row[0];

if($data){
    echo $data;
}

mysqli_close($con);
?>
```

Android Part

To experiment with this example , you need to run this on an actual device on which wifi internet is connected.

Steps	Description
1	You will use Android studio IDE to create an Android application and name it as PHPMYSQL under a package com.example.phpmysql.
2	Modify src/MainActivity.java file to add Activity code.
3	Create src/SignInActivity.java file to add PHPMYSQL code.
4	Modify layout XML file res/layout/activity_main.xml add any GUI component if required.
5	Modify res/values/string.xml file and add necessary string components.
6	Modify AndroidManifest.xml to add necessary permissions.
7	Run the application and choose a running android device and install the application on it and verify the results.

Here is the content of **src/com.example.phpmysql/MainActivity.java**.

```
package com. example. phpmysql ;

import android. app. Acti vi ty;
import android. os. Bundl e;
import android. vi ew. Vi ew;
import android. wi dget. Edi tText;
import android. wi dget. TextVi ew;
```

```
public class MainActivity extends Activity {

    private EditText usernameField, passwordField;
    private TextView status, role, method;

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

        usernameField = (EditText)findViewById(R.id.editText1);
        passwordField = (EditText)findViewById(R.id.editText2);

        status = (TextView)findViewById(R.id.textView6);
        role = (TextView)findViewById(R.id.textView7);
        method = (TextView)findViewById(R.id.textView9);
    }

    public void Login(View view){
        String username = usernameField.getText().toString();
        String password = passwordField.getText().toString();
        method.setText("Get Method");
        new SignInActivity(this, status, role, 0).execute(username, password);
    }
}
```

```
public void loginPost(View view){  
    String username = usernameField.getText().toString();  
    String password = passwordField.getText().toString();  
    method.setText("Post Method");  
    new SignInActivity(this, status, role, 1).execute(username, password);  
}  
}
```

Here is the content of **src/com.example.phpmysql/SignInActivity.java**.

```
package com.example.phpmysql;  
  
import java.io.BufferedReader;  
import java.io.InputStreamReader;  
import java.io.OutputStreamWriter;  
import java.net.URI;  
import java.net.URL;  
import java.net.URLConnection;  
import java.net.URLEncoder;  
  
import org.apache.http.HttpResponse;  
import org.apache.http.client.HttpClient;  
import org.apache.http.client.methods.HttpGet;  
import org.apache.http.impl.client.DefaultHttpClient;  
  
import android.content.Context;  
import android.os.AsyncTask;  
import android.widget.TextView;  
  
public class SignInActivity extends AsyncTask{
```



```
private TextView statusField, roleField;
private Context context;
private int byGetOrPost = 0;

//flag 0 means get and 1 means post. (By default it is get.)
public SignInActivity(Context context, TextView statusField, TextView
roleField, int flag) {
    this.context = context;
    this.statusField = statusField;
    this.roleField = roleField;
    byGetOrPost = flag;
}

protected void onPreExecute(){
}

@Override
protected String doInBackground(String... arg0) {
    if(byGetOrPost == 0){ //means by Get Method

        try{
            String username = (String)arg0[0];
            String password = (String)arg0[1];
            String link =
"http://myphpmysqlweb.hostei.com/login.php?username="+username+"&
password="+password;

            URL url = new URL(link);
            HttpClient client = new DefaultHttpClient();
            HttpGet request = new HttpGet();
```

```
request.setURI(new URI(link));
HttpResponse response = client.execute(request);
BufferedReader in = new BufferedReader(new
    InputStreamReader(response.getEntity().getContent()));

StringBuffer sb = new StringBuffer("");
String line="";

while ((line = in.readLine()) != null) {
    sb.append(line);
    break;
}

in.close();
return sb.toString();
} catch(Exception e){
    return new String("Exception: " + e.getMessage());
}
} else{
    try{
        String username = (String)arg0[0];
        String password = (String)arg0[1];

        String link="http://myphpmysqlweb.hostei.com/loginpost.php";
        String data = URLEncoder.encode("username", "UTF-8") + "=" +
            URLEncoder.encode(username, "UTF-8");
        data += "&" + URLEncoder.encode("password", "UTF-8") + "=" +
            URLEncoder.encode(password, "UTF-8");
```

```
URL url = new URL(link);
URLConnection conn = url.openConnection();

conn.setDoOutput(true);

OutputStreamWriter wr = new
OutputStreamWriter(conn.getOutputStream());

wr.write( data );
wr.flush();

BufferedReader reader = new BufferedReader(new
    InputStreamReader(conn.getInputStream()));

StringBuilder sb = new StringBuilder();
String line = null;

// Read Server Response
while((line = reader.readLine()) != null) {
    sb.append(line);
    break;
}

return sb.toString();
} catch(Exception e){
    return new String("Exception: " + e.getMessage());
}
}
```

@Override

```
protected void onPostExecute(String result){
    this.statusField.setText("Login Successful");
    this.roleField.setText(result);
}
}
```

Add the following content to **build.gradle** and rebuild the whole project.

```
android {
    useLibrary 'org.apache.http.legacy'
}
```

Here is the content of **activity_main.xml**.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <EditText
        android:id="@+id/editText2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/editText1"
        android:layout_below="@+id/editText1"
```

```
        android:layout_marginTop="25dp"
        android:ems="10"
        android:inputType="textPassword" >
</EditText>

<EditText
    android:id="@+id/editText1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentRight="true"
    android:layout_alignParentTop="true"
    android:layout_marginTop="44dp"
    android:ems="10" >

<requestFocus android:layout_width="wrap_content" />

</EditText>

<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBottom="@+id/editText1"
    android:layout_alignParentLeft="true"
    android:text="@string/Username" />

<TextView
    android:id="@+id/textView3"
    android:layout_width="wrap_content"
```

```
android:layout_height="wrap_content"
android:layout_ignoreParentTop="true"
android:layout_centerHorizontal="true"
android:text="@string/App"
android:textAppearance="?android:attr/textAppearanceLarge" />
```

<TextView

```
android:id="@+id/textView7"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignBottom="@+id/textView5"
android:layout_alignLeft="@+id/textView6"
android:text="@string/Role"
android:textAppearance="?android:attr/textAppearanceMedium"
android:textSize="10sp" />
```

<TextView

```
android:id="@+id/textView5"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_below="@+id/textView6"
android:layout_marginTop="27dp"
android:layout_toLeftOf="@+id/editText1"
android:text="@string/LoginRole" />
```

<TextView

```
android:id="@+id/textView8"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_above="@+id/textView6"
```

```
android:layout_alignLeft="@+id/textView5"
android:layout_marginBottom="27dp"
android:text="@string/method" />
```

<TextView

```
android:id="@+id/textView4"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/textView8"
android:layout_below="@+id/button1"
android:layout_marginTop="86dp"
android:text="@string/LoginStatus" />
```

<TextView

```
android:id="@+id/textView6"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignTop="@+id/textView4"
android:layout_centerHorizontal="true"
android:text="@string/Status"
android:textAppearance="?android:attr/textAppearanceMedium"
android:textSize="10sp" />
```

<TextView

```
android:id="@+id/textView9"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignBottom="@+id/textView8"
```

```
android:layout_alignLeft="@+id/textView6"
android:text="@string/Choose"
android:textAppearance="?android:attr/textAppearanceMedium"
android:textSize="10sp" />
```

<Button

```
android:id="@+id/button2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerVertical="true"
android:layout_toRightOf="@+id/textView6"
android:onClick="loginPost"
android:text="@string/LoginPost" />
```

<Button

```
android:id="@+id/button1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignBaseline="@+id/button2"
android:layout_alignBottom="@+id/button2"
android:layout_alignLeft="@+id/textView2"
android:onClick="login"
android:text="@string/LoginGet" />
```

<TextView

```
android:id="@+id/textView2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignBaseline="@+id/editText2"
```



```
        android:layout_alignBottom="@+id/editText2"
        android:layout_alignParentLeft="true"
        android:text="@string/Password" />

</RelativeLayout>
```

Here is the content of **Strings.xml**.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">PHPMYSQL</string>
    <string name="action_settings">Settings</string>
    <string name="hello_world">Hello world! </string>
    <string name="Username">Username</string>
    <string name="Password">Password</string>
    <string name="LoginGet">Login - Get</string>
    <string name="LoginPost">Login - Post</string>
    <string name="App">Login Application</string>
    <string name="LoginStatus">Login Status</string>
    <string name="LoginRole">Login Role</string>
    <string name="Status">Not Login</string>
    <string name="Role">Not assigned</string>
    <string name="method">Login Method</string>
    <string name="Choose">Choose Method</string>

</resources>
```

Here is the content of **AndroidManifest.xml**.

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.phpmysql" >

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


    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >

        <activity
            android:name="com.example.phpmysql.MainActivity"
            android:label="@string/app_name" >

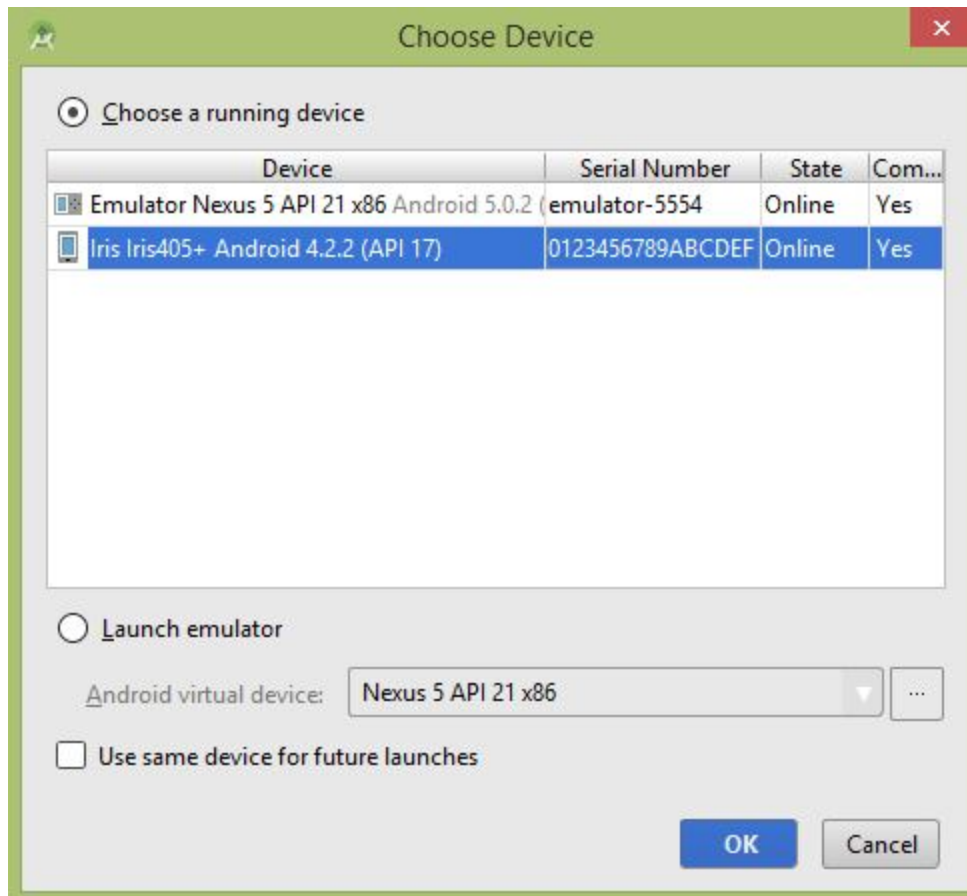
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

        </activity>


    </application>
</manifest>
```

Let's try to run your PHPMYSQL application. I assume you have connected your actual Android Mobile device with your computer. To run the app from Android studio, open one of your project's activity files and click Run  icon from the tool bar. Before starting your application, Android studio will

display following window to select an option where you want to run your Android application.



Select your mobile device as an option and then check your mobile device which will display following screen –



PHPMYSQL

Login Application

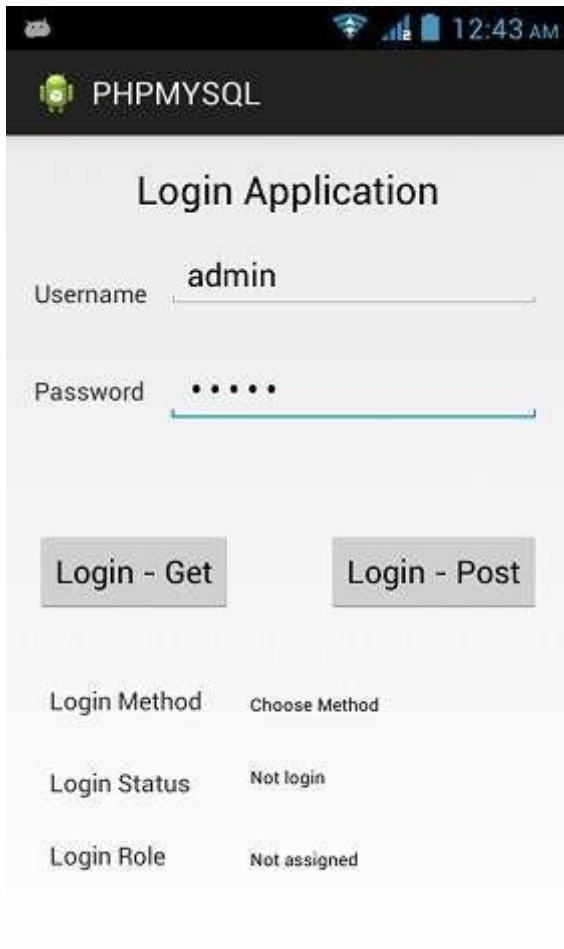
Username

Password

Login - Get Login - Post

Login Method	Choose Method
Login Status	Not login
Login Role	Not assigned

Now just type in your username and password. In my case i am typing admin as username and password. It is shown in the figure –



PHPMYSQL

Login Application

Username

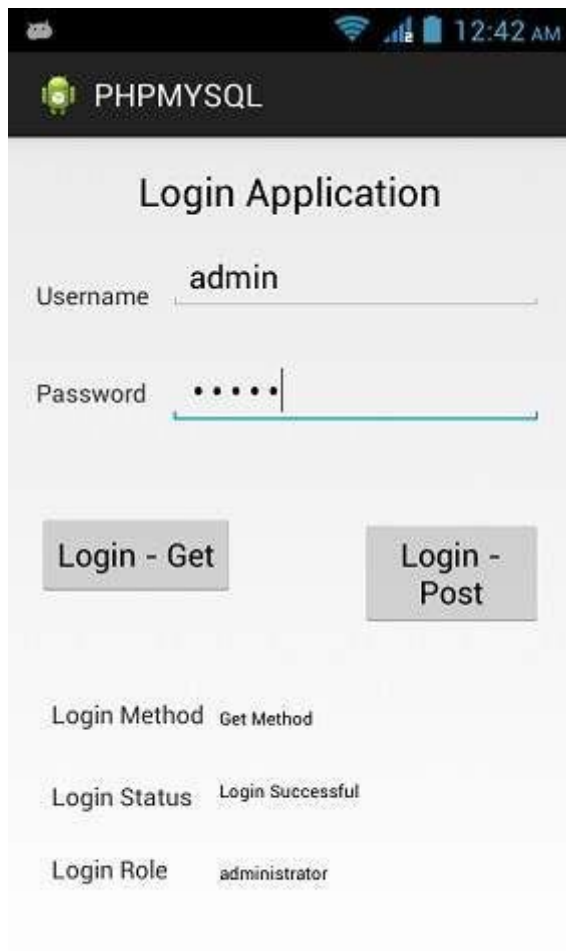
Password

Login Method Choose Method

Login Status Not login

Login Role Not assigned

Now press the Get button and wait a few seconds and response will be downloaded and will be shown to you. In this case, the response is the ROLE that is fetched in case of admin as username and password. It is shown in the figure below –



Now again press the POST button and same result would appear. It is shown in the figure below –

PHPMYSQL

Login Application

Username

Password

Login - Get Login - Post

Login Method Post Method

Login Status Login Successful

Login Role administrator

Android Progress Bar using ProgressDialog

Progress bars are used to show progress of a task. For example, when you are uploading or downloading something from the internet, it is better to show the progress of download/upload to the user.

In android there is a class called ProgressDialog that allows you to create progress bar. In order to do this, you need to instantiate an object of this class. Its syntax is.

```
ProgressDialog progress = new ProgressDialog(this);
```

Now you can set some properties of this dialog. Such as, its style, its text etc.

```
progress.setMessage("Downl oadi ng Musi c :) ");  
progress.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);  
progress.setIndeterminate(true);
```

Apart from these methods, there are other methods that are provided by the ProgressDialog class

Sr. No	Title & description
1	getMax() This method returns the maximum value of the progress.
2	incrementProgressBy(int diff)

	This method increments the progress bar by the difference of value passed as a parameter.
3	setIndeterminate(boolean indeterminate) This method sets the progress indicator as determinate or indeterminate.
4	setMax(int max) This method sets the maximum value of the progress dialog.
5	setProgress(int value) This method is used to update the progress dialog with some specific value.
6	show(Context context, CharSequence title, CharSequence message) This is a static method, used to display progress dialog.

Example

This example demonstrates the horizontal use of the progress dialog which is in fact a progress bar. It display a progress bar on pressing the button.

To experiment with this example, you need to run this on an actual device after developing the application according to the steps below.

Steps	Description
1	You will use Android studio to create an Android application under a

	package com.example.sairamkrishna.myapplication.
2	Modify src/MainActivity.java file to add progress code to display the progress dialog.
3	Modify res/layout/activity_main.xml file to add respective XML code.
4	Run the application and choose a running android device and install the application on it and verify the results.

Following is the content of the modified main activity file **src/MainActivity.java**.

```
package com. exampl e. sai ramkri shna. myappl i cati on;

import android. app. ProgressDi al og;
import android. support. v7. app. Acti onBarActi vi ty;
import android. os. Bundl e;
import android. vi ew. Vi ew;
import android. wi dget. Button;

publ ic cl ass Mai nActi vi ty extends Acti onBarActi vi ty {
    Button b1;
    pri vate ProgressDi al og progress;

    protecte d voi d onCreate(Bundl e savedI nstanceState) {
        super. onCreate(savedI nstanceState);
        setConte ntVi ew(R. layout. acti vi ty_mai n);
        b1 = (Button) fi ndVi ewByI d(R. i d. button2);
    }
}
```

```
public void download(View view){
    progress=new ProgressDialog(this);
    progress.setMessage("Downl oadi ng Musi c");
    progress.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
    progress.setIndeterminate(true);
    progress.setProgress(0);
    progress.show();

    final int totalTime = 100;
    final Thread t = new Thread() {
        @Override
        public void run() {
            int jumpTime = 0;

            while(jumpTime < totalTime) {
                try {
                    sleep(200);
                    jumpTime += 5;
                    progress.setProgress(jumpTime);
                } catch (InterruptedException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
        }
    };
    t.start();
}
```

```
}
```

Modify the content of **res/layout/activity_main.xml** to the following –

```
<RelativeLayout 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:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/textView"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:textSize="30dp"
        android:text="Progress bar" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Tutorial's Point"
        android:id="@+id/textView2"
        android:layout_below="@+id/textView"
        android:layout_centerHorizontal="true"
        android:textSize="35dp"
```

```
        android:textColor="#ff16ff01" />

        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Download"
            android:onClick="download"
            android:id="@+id/button2"
            android:layout_marginLeft="125dp"
            android:layout_marginStart="125dp"
            android:layout_centerVertical="true" />

    </RelativeLayout>
```

This is the default **AndroidManifest.xml** –

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.sai ramkri shna.myappl ication" >
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >


        <activity
            android:name=". MainActivity"
            android:label="@string/app_name" >

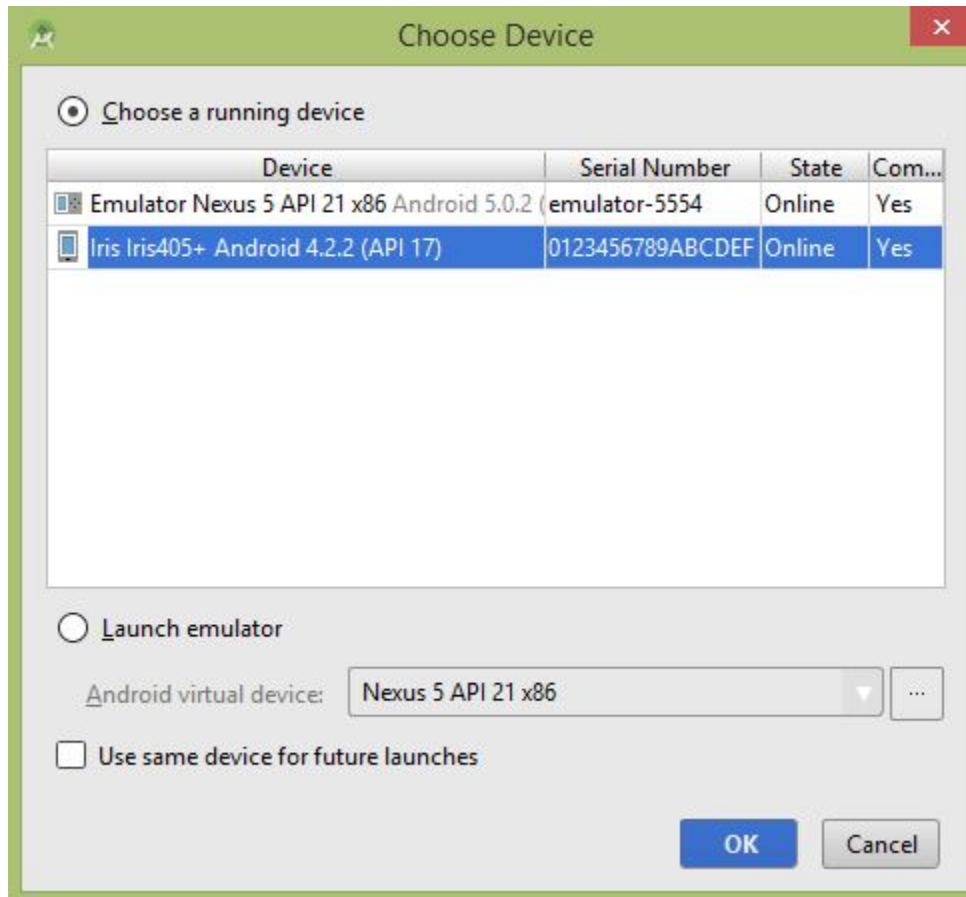
            <intent-filter>
```

```
<action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>

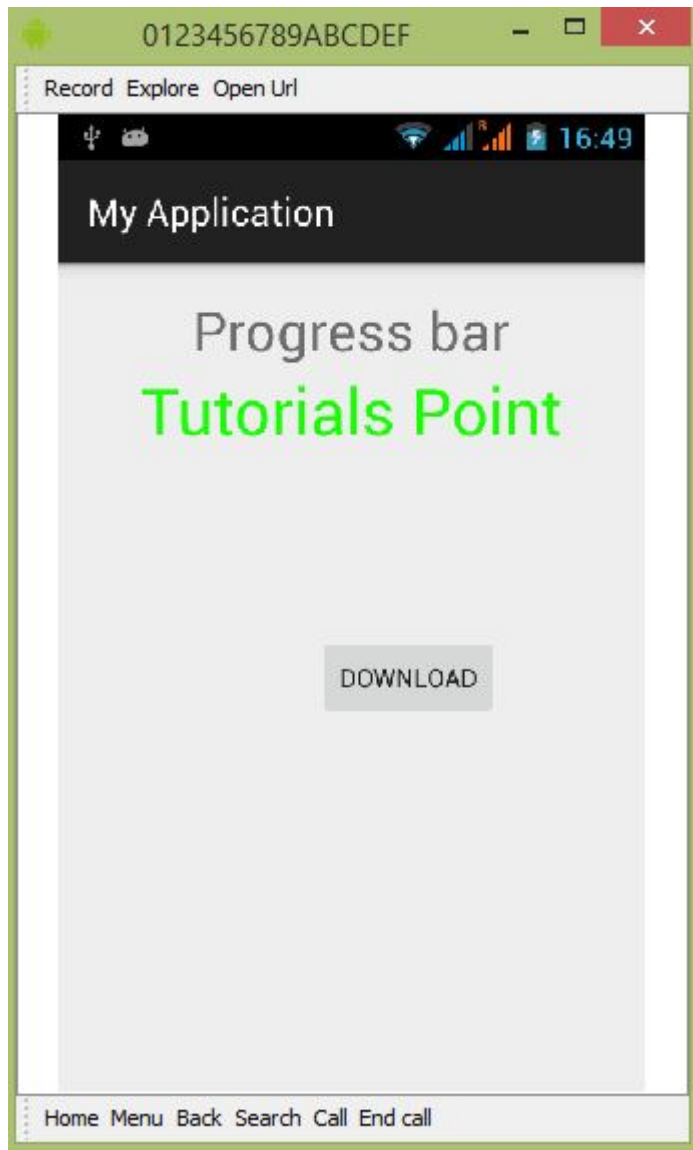
</activity>

</application>
</manifest>
```

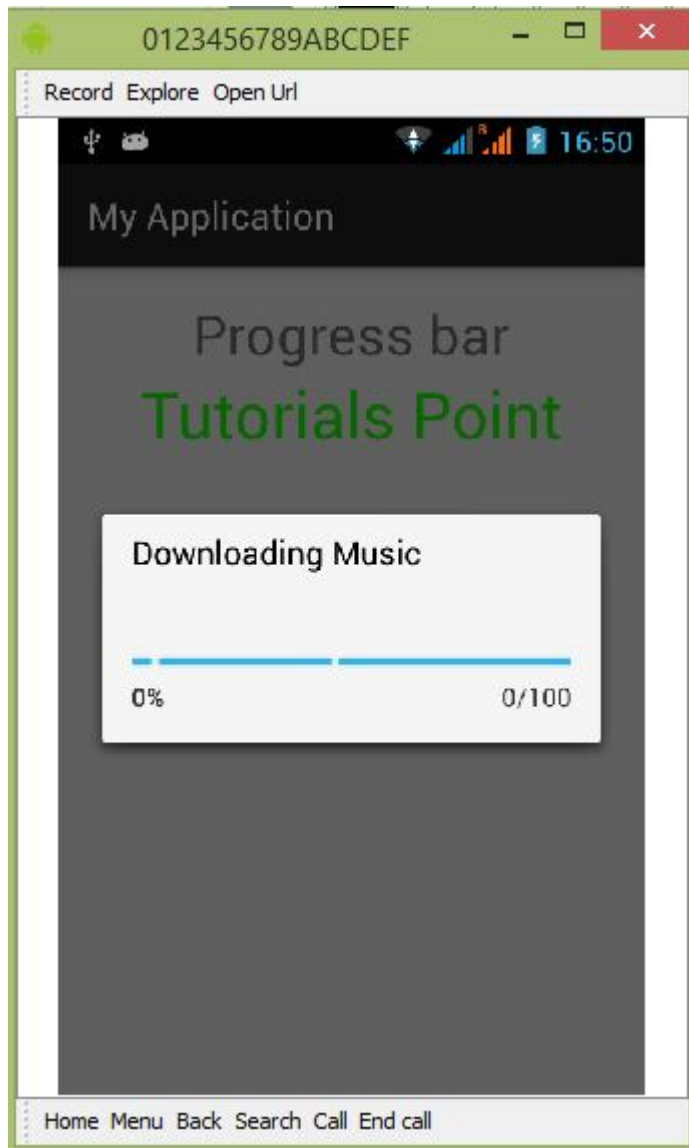
Let's try to run your application. We assume, you have connected your actual Android Mobile device with your computer. To run the app from Android studio, open one of your project's activity files and click Run  icon from the toolbar. Before starting your application, Android studio will display following window to select an option where you want to run your Android application.



Select your mobile device as an option and then check your mobile device which will display following screen –



Just press the button to start the Progress bar. After pressing, following screen would appear –



It will continuously update itself.

Android - Google Maps

Android allows us to integrate google maps in our application. You can show any location on the map , or can show different routes on the map e.t.c. You can also customize the map according to your choices.

Google Map - Layout file

Now you have to add the map fragment into xml layout file. Its syntax is given below –

```
<fragment
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.MapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
```

Google Map - AndroidManifest file

The next thing you need to do is to add some permissions along with the Google Map API key in the AndroidManifest.XML file. Its syntax is given below –

```
<!--Permissions-->

<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

<!--Google MAP API key-->
```

```
<meta-data
    android:name="com.google.android.maps.v2.API_KEY"
    android:value="AIzaSyDKymeBXNeiFWY5jRUejv6zItpmr2MVyQ0" />
```

Customizing Google Map

You can easily customize google map from its default view , and change it according to your demand.

Adding Marker

You can place a maker with some text over it displaying your location on the map. It can be done by via **addMarker()** method. Its syntax is given below –

```
final LatLng TutorialSPoint = new LatLng(21 , 57);
Marker TP = googleMap.addMarker(new MarkerOptions()
    .position(TutorialSPoint).title("TutorialSPoint"));
```

Channing Map Type

You can also change the type of the MAP. There are four different types of map and each give different view of the map. These types are Normal,Hybrid,Satellite and terrain. You can use them as below

```
googleMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
googleMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);
googleMap.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
googleMap.setMapType(GoogleMap.MAP_TYPE_TERRAIN);
```

Enable/Disable zoom

You can also enable or disable the zoom gestures in the map by calling the **setZoomControlsEnabled(boolean)** method. Its syntax is given below –

```
googleMap.getUiSettings().setZoomGesturesEnabled(true);
```

Apart from these customization, there are other methods available in the GoogleMap class , that helps you more customize the map. They are listed below –

Sr.No	Method & description
1	addCircle(CircleOptions options) This method add a circle to the map
2	addPolygon(PolygonOptions options) This method add a polygon to the map
3	addTileOverlay(TileOverlayOptions options) This method add tile overlay to the map
4	animateCamera(CameraUpdate update) This method Moves the map according to the update with an animation
5	clear() This method removes everything from the map.
6	getMyLocation() This method returns the currently displayed user location.

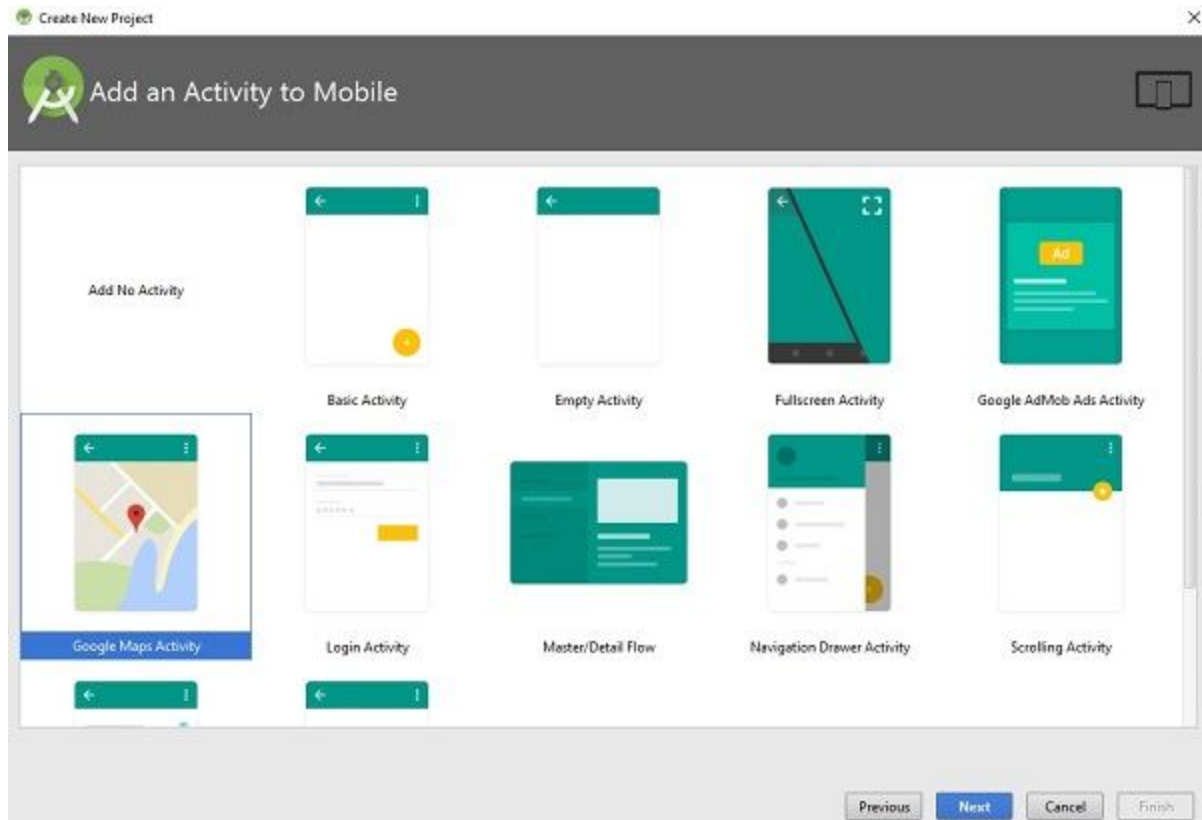
7	moveCamera(CameraUpdate update) This method repositions the camera according to the instructions defined in the update
8	setTrafficEnabled(boolean enabled) This method Toggles the traffic layer on or off.
9	snapshot(GoogleMap.SnapshotReadyCallback callback) This method Takes a snapshot of the map
10	stopAnimation() This method stops the camera animation if there is one in progress

Example

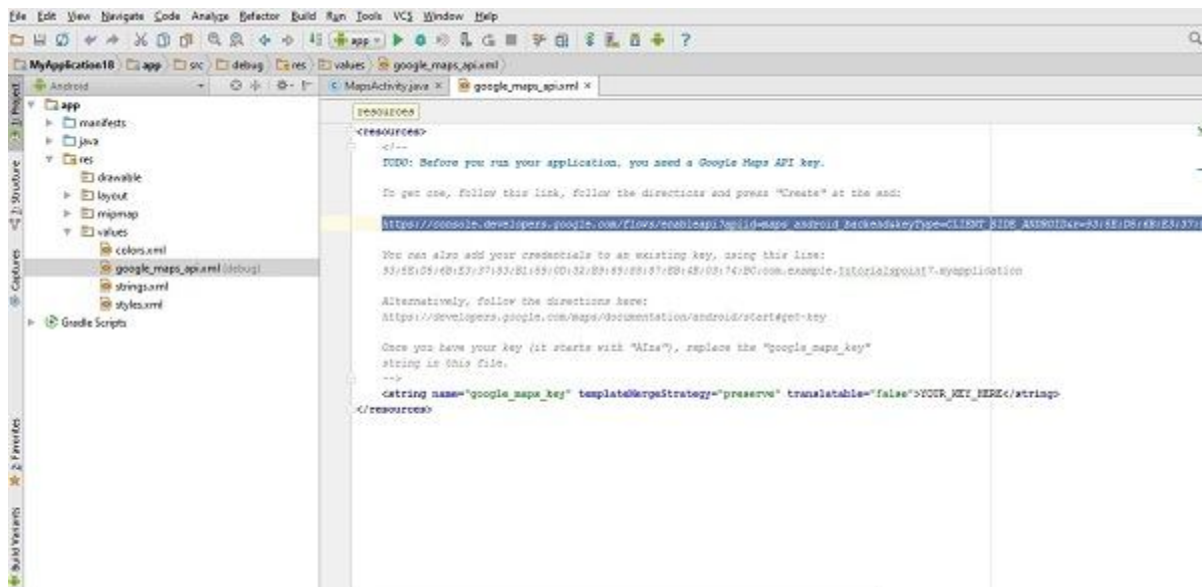
Here is an example demonstrating the use of GoogleMap class. It creates a basic M application that allows you to navigate through the map.

To experiment with this example , you can run this on an actual device or in an emulator.

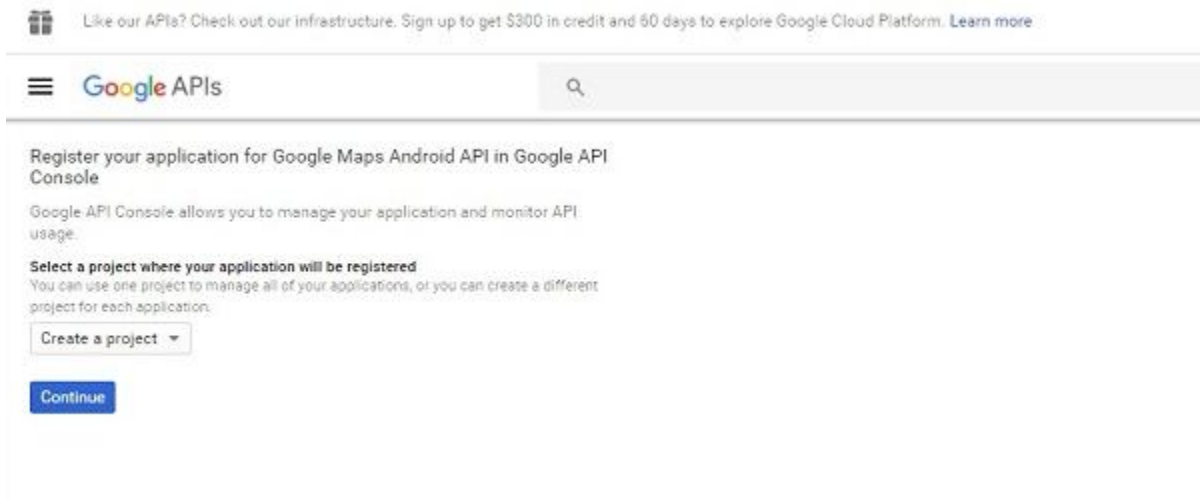
Create a project with google maps activity as shown below –



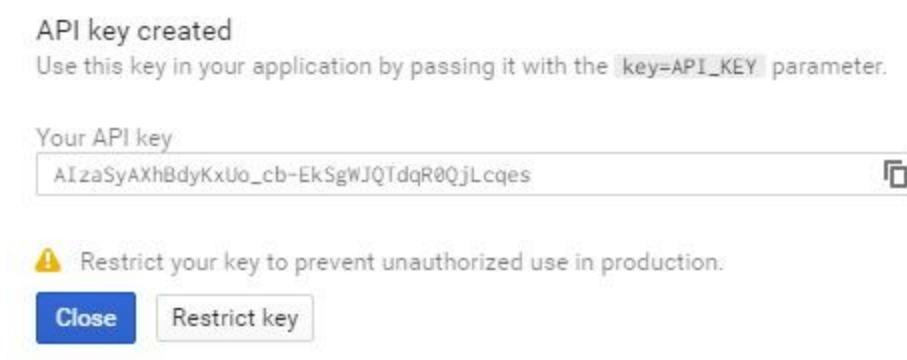
It will open the following screen and copy the console url for API Key as shown below –



Copy this and paste it to your browser. It will give the following screen –



Click on continue and click on Create API Key then it will show the following screen



Here is the content of **activity_main.xml**.

```
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.tutorialspoint7.myapplication.MapsActivity" />
```

Here is the content of **MapActivity.java**.

In the below code we have given sample latitude and longitude details

```
package com.example.tutorialspoint7.myapplication;

import android.support.v4.app.FragmentActivity;
import android.os.Bundle;

import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;

public class MapsActivity extends FragmentActivity implements
    OnMapReadyCallback {

    private GoogleMap mMap;

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

        // Obtain the SupportMapFragment and get notified when the map is ready
        to be used.

        SupportMapFragment mapFragment = (SupportMapFragment)
        getSupportFragmentManager()
            .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }

    /**
```



```
* Manipulates the map once available.
* This callback is triggered when the map is ready to be used.
* This is where we can add markers or lines, add listeners or move the
camera.
* In this case, we just add a marker near Sydney, Australia.
* If Google Play services is not installed on the device.
* This method will only be triggered once the user has installed
  Google Play services and returned to the app.
*/

@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;
    // Add a marker in Sydney and move the camera
    LatLng TutorialSPoint = new LatLng(21, 57);
    mMap.addMarker(new
MarkerOptions().position(TutorialSPoint).title("Tutorial spoint.com"));
    mMap.moveCamera(CameraUpdateFactory.newLatLng(TutorialSPoint));
}
}
```

Following is the content of **AndroidManifest.xml** file.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.tutorialspoint7.myapplication">

    <!--
        The ACCESS_COARSE/FINE_LOCATION permissions are not required to use
        Google Maps Android API v2, but you must specify either coarse or fine
```

Location permissions for the 'MyLocation' functionality.

-->

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

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

```
<application
```

```
    android:allowBackup="true"
```

```
    android:icon="@mipmap/ic_launcher"
```

```
    android:label="@string/app_name"
```

```
    android:supportRtl="true"
```

```
    android:theme="@style/AppTheme">
```

```
<!--
```

The API key for Google Maps-based APIs is defined as a string resource.

(See the file "res/values/google_maps_api.xml").

Note that the API key is linked to the encryption key used to sign the APK.

You need a different API key for each encryption key, including the release key

that is used to sign the APK for publishing.

You can define the keys for the debug and

release targets in src/debug/ and src/release/.

-->

```
<meta-data
```

```
    android:name="com.google.android.geo.API_KEY"
```

```
    android:value="AIzaSyAXhBdyKxUo_cb-EkSgWJQTdqR0QjLcques" />
```

```
<acti vi ty
    android: name=". MapsActi vi ty"
    android: label="@string/ti tle_acti vi ty_maps">
    <i ntent-fi lter>
        <acti on android: name="android. i ntent. acti on. MAIN" />
        <category android: name="android. i ntent. category. LAUNCHER" />
    </i ntent-fi lter>
</acti vi ty>
</appl i cati on>

</mani fest>
```

Android - Sending SMS

Advertisements

[Previous Page](#)

[Next Page](#) _

In Android, you can use SmsManager API or devices Built-in SMS application to send SMS's. In this tutorial, we shows you two basic examples to send SMS message –

SmsManager API

```
SmsManager smsManager = SmsManager.getDefault();  
smsManager.sendTextMessage("phoneNo", null, "sms message", null, null);
```

Built-in SMS application

```
Intent sendIntent = new Intent(Intent.ACTION_VIEW);  
sendIntent.putExtra("sms_body", "default content");  
sendIntent.setType("vnd.android-dir/mms-sms");  
startActivity(sendIntent);
```

Of course, both need **SEND_SMS permission**.

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

Apart from the above method, there are few other important functions available in SmsManager class. These methods are listed below –

Sr.No.	Method & Description
1	ArrayList<String> divideMessage(String text) This method divides a message text into several fragments, none bigger than the maximum SMS message size.

2	static SmsManager getDefault() This method is used to get the default instance of the SmsManager
3	void sendDataMessage(String destinationAddress, String scAddress, short destinationPort, byte[] data, PendingIntent sentIntent, PendingIntent deliveryIntent) This method is used to send a data based SMS to a specific application port.
4	void sendMultipartTextMessage(String destinationAddress, String scAddress, ArrayList<String> parts, ArrayList<PendingIntent> sentIntents, ArrayList<PendingIntent> deliveryIntents) Send a multi-part text based SMS.
5	void sendTextMessage(String destinationAddress, String scAddress, String text, PendingIntent sentIntent, PendingIntent deliveryIntent) Send a text based SMS.

Example

Following example shows you in practical how to use SmsManager object to send an SMS to the given mobile number.

To experiment with this example, you will need actual Mobile device equipped with latest Android OS, otherwise you will have to struggle with emulator which may not work.

Step	Description
------	-------------

1	You will use Android Studio IDE to create an Android application and name it as <i>tutorialspoint</i> under a package <i>com.example.tutorialspoint</i> .
2	Modify <i>src/MainActivity.java</i> file and add required code to take care of sending sms.
3	Modify layout XML file <i>res/layout/activity_main.xml</i> add any GUI component if required. I'm adding a simple GUI to take mobile number and SMS text to be sent and a simple button to send SMS.
4	No need to define default string constants at <i>res/values/strings.xml</i> . Android studio takes care of default constants.
5	Modify <i>AndroidManifest.xml</i> as shown below
6	Run the application to launch Android emulator and verify the result of the changes done in the application.

Following is the content of the modified main activity file **src/com.example.tutorialspoint/MainActivity.java**.

```
package com.example.tutorialspoint;

import android.Manifest;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.app.Activity;

import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.telephony.SmsManager;
```

```
import android.util.Log;
import android.view.Menu;
import android.view.View;

import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends Activity {
    private static final int MY_PERMISSIONS_REQUEST_SEND_SMS =0 ;
    Button sendBtn;
    EditText txtphoneNo;
    EditText txtMessage;
    String phoneNo;
    String message;

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

        sendBtn = (Button) findViewById(R.id.btnSendSMS);
        txtphoneNo = (EditText) findViewById(R.id.editText);
        txtMessage = (EditText) findViewById(R.id.editText2);

        sendBtn.setOnClickListener(new View.OnClickListener() {
            public void onClick(View view) {
                sendSMSMessage();
            }
        })
    }
}
```

```
});  
}  
  
protected void sendSMSMessage() {  
    phoneNo = txtphoneNo.getText().toString();  
    message = txtMessage.getText().toString();  
  
    if (ContextCompat.checkSelfPermission(this,  
        Manifest.permission.SEND_SMS)  
        != PackageManager.PERMISSION_GRANTED) {  
        if (ActivityCompat.shouldShowRequestPermissionRationale(this,  
            Manifest.permission.SEND_SMS)) {  
        } else {  
            ActivityCompat.requestPermissions(this,  
                new String[]{Manifest.permission.SEND_SMS},  
                MY_PERMISSIONS_REQUEST_SEND_SMS);  
        }  
    }  
}  
  
@Override  
public void onRequestPermissionsResult(int requestCode, String  
permissions[], int[] grantResults) {  
    switch (requestCode) {  
        case MY_PERMISSIONS_REQUEST_SEND_SMS: {  
            if (grantResults.length > 0  
                && grantResults[0] == PackageManager.PERMISSION_GRANTED) {  
                SmsManager smsManager = SmsManager.getDefault();  
                smsManager.sendTextMessage(phoneNo, null, message, null,  
null);  
            }  
        }  
    }  
}
```



```
        Toast.makeText(getApplicationContext(), "SMS sent.",
            Toast.LENGTH_LONG).show();
    } else {
        Toast.makeText(getApplicationContext(),
            "SMS failed, please try again.", Toast.LENGTH_LONG).show();
        return;
    }
}
}
```

Following will be the content of **res/layout/activity_main.xml** file –

Here abc indicates about tutorialspoint logo

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="MainActivity">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
```

```
android:layout_height="wrap_content"
android:text="Sending SMS Example"
android:layout_alignParentTop="true"
android:layout_centerHorizontal="true"
android:textSize="30dp" />
```

<TextView

```
android:id="@+id/textView2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Tutorials point "
android:textColor="#ff87ff09"
android:textSize="30dp"
android:layout_below="@+id/textView1"
android:layout_alignRight="@+id/imageButton"
android:layout_alignEnd="@+id/imageButton" />
```

<ImageButton

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:id="@+id/imageButton"
android:src="@drawable/abc"
android:layout_below="@+id/textView2"
android:layout_centerHorizontal="true" />
```

<EditText

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:id="@+id/editText"
```

```
android:hint="Enter Phone Number"
android:phoneNumber="true"
android:textColorHint="@color/abc_primary_text_material_dark"
android:layout_below="@+id/imageButton"
android:layout_centerHorizontal="true" />
```

<EditText

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:id="@+id/editText2"
android:layout_below="@+id/editText"
android:layout_alignLeft="@+id/editText"
android:layout_alignStart="@+id/editText"
android:textColorHint="@color/abc_primary_text_material_dark"
android:layout_alignRight="@+id/imageButton"
android:layout_alignEnd="@+id/imageButton"
android:hint="Enter SMS" />
```

<Button

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Send Sms"
android:id="@+id/btnSendSMS"
android:layout_below="@+id/editText2"
android:layout_centerHorizontal="true"
android:layout_marginTop="48dp" />
```

</RelativeLayout>

Following will be the content of **res/values/strings.xml** to define two new constants –

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">tutorialspoint</string>
</resources>
```

Following is the default content of **AndroidManifest.xml** –

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


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

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >

        <activity
            android:name="com.example.tutorialspoint.MainActivity"
            android:label="@string/app_name" >

            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
```

```
</acti vi ty>  
  
</appl i cati on>  
</mani fest>
```

Let's try to run your **tutorialspoint** application. I assume you have connected your actual Android Mobile device with your computer. To run the app from Android studio, open one of your project's activity files and click Run  icon from the toolbar. Before starting your application, Android studio installer will display following window to select an option where you want to run your Android application.



Now you can enter a desired mobile number and a text message to be sent on that number. Finally click on **Send SMS** button to send your SMS. Make

sure your GSM/CDMA connection is working fine to deliver your SMS to its recipient.

You can take a number of SMS separated by comma and then inside your program you will have to parse them into an array string and finally you can use a loop to send message to all the given numbers. That's how you can write your own SMS client. Next section will show you how to use existing SMS client to send SMS.

Using Built-in Intent to send SMS

You can use Android Intent to send SMS by calling built-in SMS functionality of the Android. Following section explains different parts of our Intent object required to send an SMS.

Intent Object - Action to send SMS

You will use **ACTION_VIEW** action to launch an SMS client installed on your Android device. Following is simple syntax to create an intent with ACTION_VIEW action.

```
Intent smsIntent = new Intent(Intent.ACTION_VIEW);
```

Intent Object - Data/Type to send SMS

To send an SMS you need to specify **smsto:** as URI using setData() method and data type will be to **vnd.android-dir/mms-sms** using setType() method as follows –

```
smsIntent.setData(Uri.parse("smsto:"));  
smsIntent.setType("vnd.android-dir/mms-sms");
```

Intent Object - Extra to send SMS

Android has built-in support to add phone number and text message to send an SMS as follows –

```
smsIntent.putExtra("address" , new String("0123456789;3393993300"));  
smsIntent.putExtra("sms_body" , "Test SMS to Angila");
```

Here address and sms_body are case sensitive and should be specified in small characters only. You can specify more than one number in single string but separated by semi-colon (;).

Example

Following example shows you in practical how to use Intent object to launch SMS client to send an SMS to the given recipients.

To experiment with this example, you will need actual Mobile device equipped with latest Android OS, otherwise you will have to struggle with emulator which may not work.

Step	Description
1	You will use Android studio IDE to create an Android application and name it as <i>tutorialspoint</i> under a package <i>com.example.tutorialspoint</i> .
2	Modify <i>src/MainActivity.java</i> file and add required code to take care of sending SMS.
3	Modify layout XML file <i>res/layout/activity_main.xml</i> add any GUI component if required. I'm adding a simple button to launch SMS Client.
4	No need to define default constants.Android studio takes care of default constants.
5	Modify <i>AndroidManifest.xml</i> as shown below
6	Run the application to launch Android emulator and verify the result of the changes done in the application.

Following is the content of the modified main activity file **src/com.example.tutorialspoint/MainActivity.java**.

```
package com. exampl e. tutori al spoi nt;
```

```
import android.net.Uri;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.util.Log;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Button startBtn = (Button) findViewById(R.id.button);
        startBtn.setOnClickListener(new View.OnClickListener() {
            public void onClick(View view) {
                sendSMS();
            }
        });
    }

    protected void sendSMS() {
        Log.i("Send SMS", "");
        Intent smsIntent = new Intent(Intent.ACTION_VIEW);
```



```
        smsIntent.setData(Uri.parse("smsto:"));
        smsIntent.setType("vnd.android-dir/mms-sms");
        smsIntent.putExtra("address" , new String ("01234"));
        smsIntent.putExtra("sms_body" , "Test ");

        try {
            startActivity(smsIntent);
            finish();
            Log.i("Finished sending SMS...", "");
        } catch (android.content.ActivityNotFoundException ex) {
            Toast.makeText(MainActivity.this,
                "SMS failed, please try again later.", Toast.LENGTH_SHORT).show();
        }
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
        // present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

Following will be the content of **res/layout/activity_main.xml** file –

Here abc indicates about tutorialspoint logo

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
android:paddingBottom="@dimen/activity_vertical_margin"
tools:context=".MainActivity">
```

<TextView

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Drag and Drop Example"
android:id="@+id/textView"
android:layout_alignParentTop="true"
android:layout_centerHorizontal="true"
android:textSize="30dp" />
```

<TextView

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Tutorials Point "
android:id="@+id/textView2"
android:layout_below="@+id/textView"
android:layout_centerHorizontal="true"
android:textSize="30dp"
android:textColor="#ff14be3c" />
```

<ImageView

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
```

```
android:id="@+id/imageView"
android:src="@drawable/abc"
android:layout_marginTop="48dp"
android:layout_below="@+id/textView2"
android:layout_centerHorizontal="true" />
```

<Button

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Compose SMS"
android:id="@+id/button"
android:layout_below="@+id/imageView"
android:layout_alignRight="@+id/textView2"
android:layout_alignEnd="@+id/textView2"
android:layout_marginTop="54dp"
android:layout_alignLeft="@+id/imageView"
android:layout_alignStart="@+id/imageView" />
```

</RelativeLayout>

Following will be the content of **res/values/strings.xml** to define two new constants –

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">tutorial spoit</string>
</resources>
```

Following is the default content of **AndroidManifest.xml** –

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.tutorialspoint" >


    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >

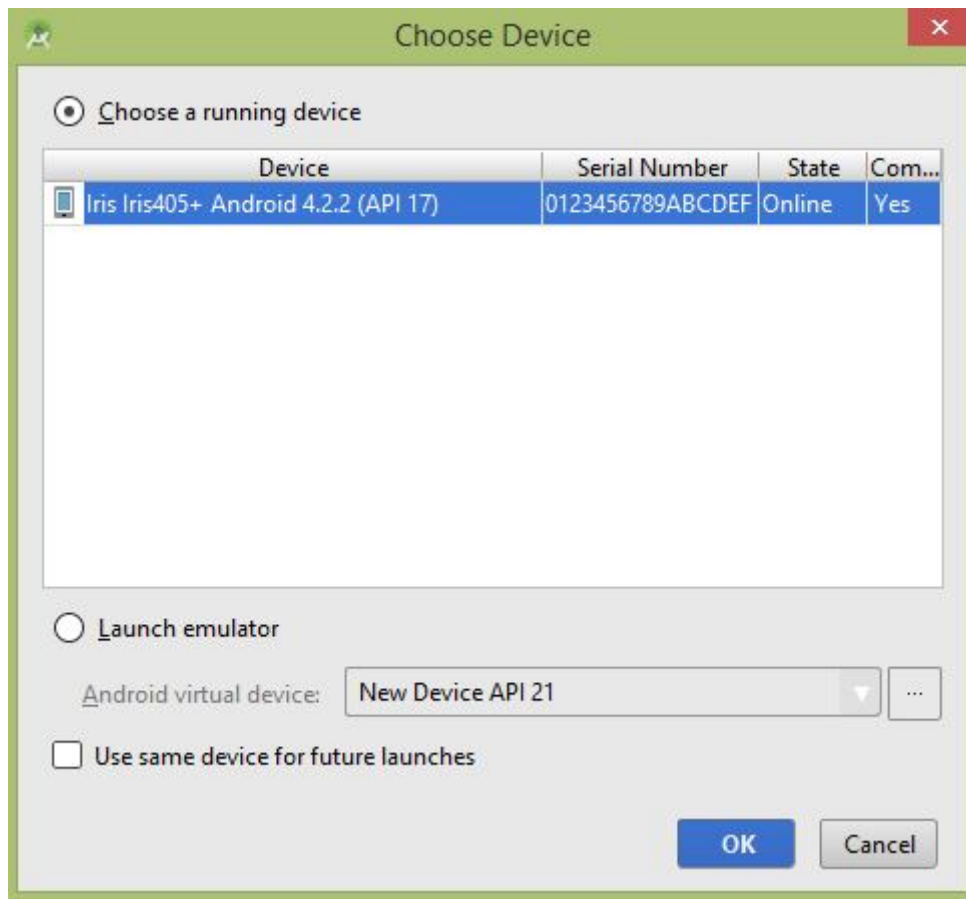
        <activity
            android:name="com.example.tutorialspoint.MainActivity"
            android:label="@string/app_name" >

                <intent-filter>
                    <action android:name="android.intent.action.MAIN" />
                    <category android:name="android.intent.category.LAUNCHER" />
                </intent-filter>

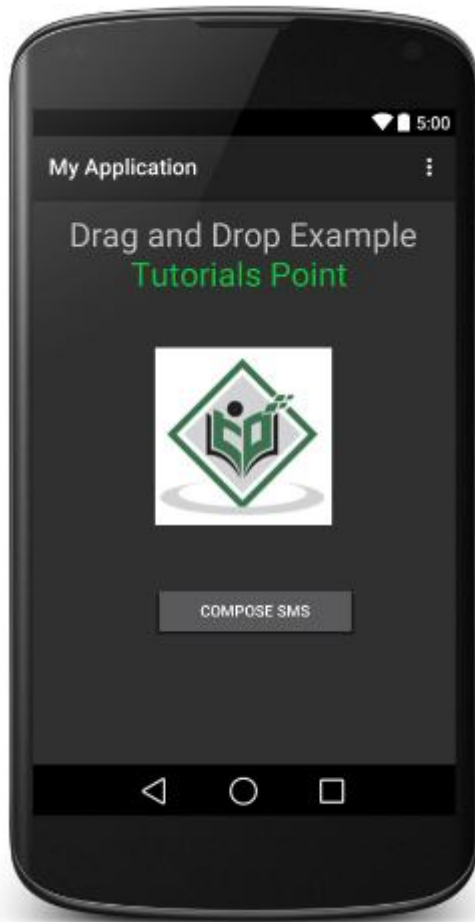
            </activity>

        </application>
    </manifest>
```

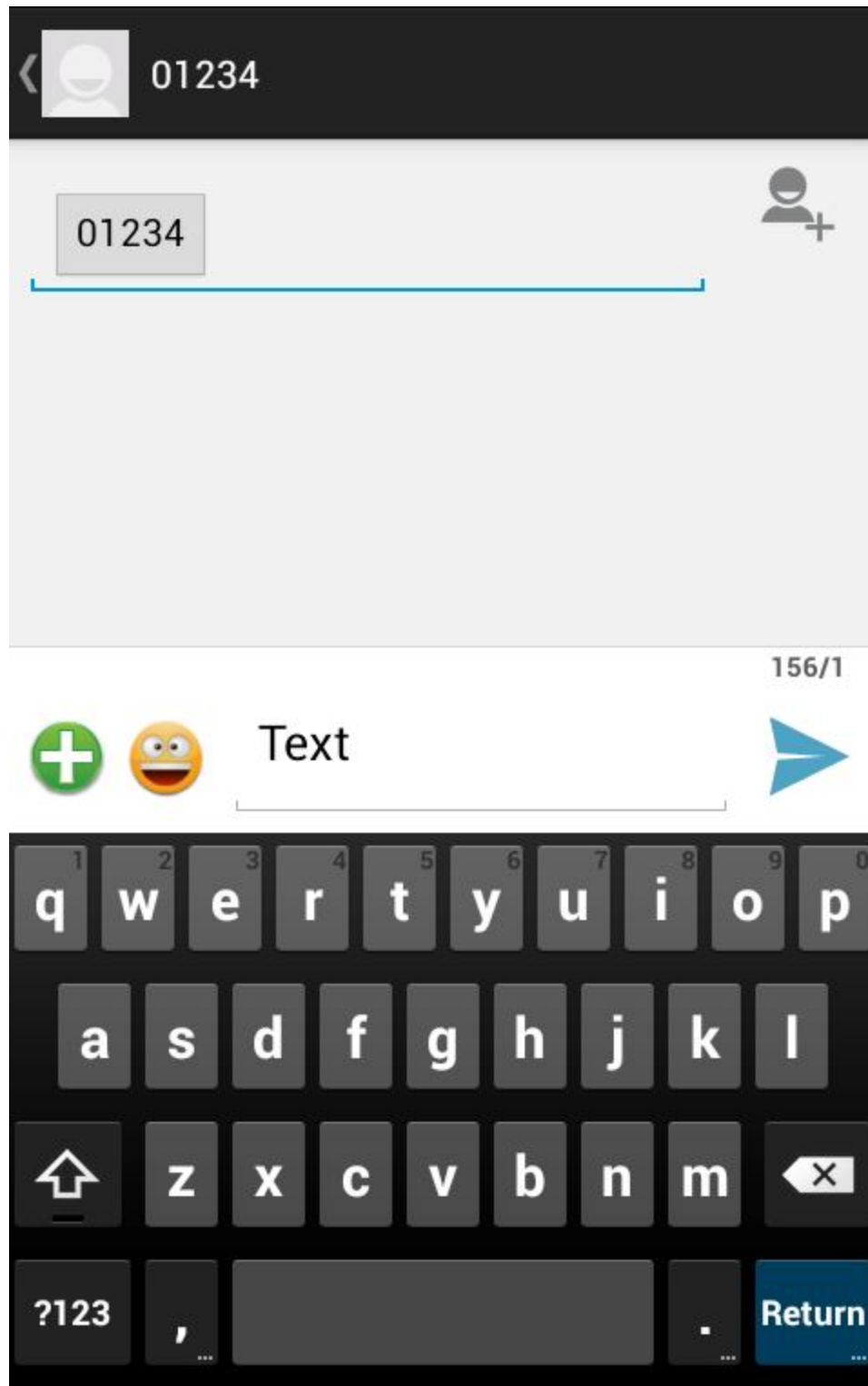
Let's try to run your **tutorialspoint** application. I assume you have connected your actual Android Mobile device with your computer. To run the app from Android studio, open one of your project's activity files and click Run  icon from the toolbar. Before starting your application, Android studio will display following window to select an option where you want to run your Android application.



Select your mobile device as an option and then check your mobile device which will display following screen –



Now use **Compose SMS** button to launch Android built-in SMS clients which is shown below –



You can modify either of the given default fields and finally use send SMS button to send your SMS to the mentioned recipient.