Week 9: Develop a native application that uses GPS location information.

To develop a native Android application that uses GPS location information, you'll need to handle permissions, access the device's GPS, and manage location updates. Here's a step-by-step guide on how to create a simple Android app that retrieves and displays the user's current location using GPS:

Create Android Application

Step	Description
1	You will use Android studio IDE to create an Android application and name it as *Tutorialspoint* under a package *com.example.tutorialspoint7.myapplication*.
2	add *src/GPSTracker.java* file and add required code.
3	Modify *src/MainActivity.java* file and add required code as shown below to take care of getting current location and its equivalent address.
4	Modify layout XML file *res/layout/activity_main.xml* to add all GUI components which include three buttons and two text views to show location/address.
5	Modify *res/values/strings.xml* to define required constant values
6	Modify *AndroidManifest.xml* as shown below
7	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 **MainActivity.java**.

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

import android.Manifest;
import android.app.Activity;
import android.os.Bundle;
import android.support.v4.app.ActivityCompat;
import android.test.mock.MockPackageManager;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

public class MainActivity extends Activity {

   Button btnShowLocation;
   private static final int REQUEST_CODE_PERMISSION = 2;
   String mPermission = Manifest.permission.ACCESS_FINE_LOCATION;

   // GPSTracker class
```

```
   GPSTracker gps;

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

      try {
         if (ActivityCompat.checkSelfPermission(this, mPermission)
            != MockPackageManager.PERMISSION_GRANTED) {

            ActivityCompat.requestPermissions(this, new
String[]{mPermission},
               REQUEST_CODE_PERMISSION);

            // If any permission above not allowed by user, this condition
will
               execute every time, else your else part will work
         }
      } catch (Exception e) {
         e.printStackTrace();
      }

      btnShowLocation = (Button) findViewById(R.id.button);

      // show location button click event
      btnShowLocation.setOnClickListener(new View.OnClickListener() {

         @Override
         public void onClick(View arg0) {
            // create class object
            gps = new GPSTracker(MainActivity.this);

            // check if GPS enabled
            if(gps.canGetLocation()){

               double latitude = gps.getLatitude();
               double longitude = gps.getLongitude();

               // \n is for new line
               Toast.makeText(getApplicationContext(), "Your Location is -
\nLat: "
                  + latitude + "\nLong: " + longitude,
Toast.LENGTH_LONG).show();
            }else{
               // can't get location
               // GPS or Network is not enabled
               // Ask user to enable GPS/network in settings
               gps.showSettingsAlert();
            }

         }
      });
   }
}
```

Following is the content of the modified main activity file **GPSTracker.java**.

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

import android.app.AlertDialog;
import android.app.Service;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.os.IBinder;
import android.provider.Settings;
import android.util.Log;

public class GPSTracker extends Service implements LocationListener {

   private final Context mContext;

   // flag for GPS status
   boolean isGPSEnabled = false;

   // flag for network status
   boolean isNetworkEnabled = false;

   // flag for GPS status
   boolean canGetLocation = false;

   Location location; // location
   double latitude; // latitude
   double longitude; // longitude

   // The minimum distance to change Updates in meters
   private static final long MIN_DISTANCE_CHANGE_FOR_UPDATES = 10; // 10
meters

   // The minimum time between updates in milliseconds
   private static final long MIN_TIME_BW_UPDATES = 1000 * 60 * 1; // 1 minute

   // Declaring a Location Manager
   protected LocationManager locationManager;

   public GPSTracker(Context context) {
      this.mContext = context;
      getLocation();
   }

   public Location getLocation() {
      try {
         locationManager = (LocationManager)
mContext.getSystemService(LOCATION_SERVICE);

         // getting GPS status
```

```
         isGPSEnabled =
locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER);

         // getting network status
         isNetworkEnabled = locationManager
            .isProviderEnabled(LocationManager.NETWORK_PROVIDER);

         if (!isGPSEnabled && !isNetworkEnabled) {
            // no network provider is enabled
         } else {
            this.canGetLocation = true;
            // First get location from Network Provider
            if (isNetworkEnabled) {
               locationManager.requestLocationUpdates(
                  LocationManager.NETWORK_PROVIDER,
                  MIN_TIME_BW_UPDATES,
                  MIN_DISTANCE_CHANGE_FOR_UPDATES, this);

               Log.d("Network", "Network");
               if (locationManager != null) {
                  location = locationManager
                     .getLastKnownLocation(LocationManager.NETWORK_PROVIDER);

                  if (location != null) {
                     latitude = location.getLatitude();
                     longitude = location.getLongitude();
                  }
               }
            }

            // if GPS Enabled get lat/long using GPS Services
            if (isGPSEnabled) {
               if (location == null) {
                  locationManager.requestLocationUpdates(
                     LocationManager.GPS_PROVIDER,
                     MIN_TIME_BW_UPDATES,
                     MIN_DISTANCE_CHANGE_FOR_UPDATES, this);

                  Log.d("GPS Enabled", "GPS Enabled");
                  if (locationManager != null) {
                     location = locationManager
                        .getLastKnownLocation(LocationManager.GPS_PROVIDER);

                     if (location != null) {
                        latitude = location.getLatitude();
                        longitude = location.getLongitude();
                     }
                  }
               }
            }
         }

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

      return location;
```

```
    }

    /**
       * Stop using GPS listener
       * Calling this function will stop using GPS in your app
    * */

    public void stopUsingGPS(){
       if(locationManager != null){
          locationManager.removeUpdates(GPSTracker.this);
       }
    }

    /**
       * Function to get latitude
    * */

    public double getLatitude(){
       if(location != null){
          latitude = location.getLatitude();
       }

       // return latitude
       return latitude;
    }

    /**
       * Function to get longitude
    * */

    public double getLongitude(){
       if(location != null){
          longitude = location.getLongitude();
       }

       // return longitude
       return longitude;
    }

    /**
       * Function to check GPS/wifi enabled
       * @return boolean
    * */

    public boolean canGetLocation() {
       return this.canGetLocation;
    }

    /**
       * Function to show settings alert dialog
       * On pressing Settings button will lauch Settings Options
    * */

    public void showSettingsAlert(){
       AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext);

       // Setting Dialog Title
```

```
      alertDialog.setTitle("GPS is settings");

      // Setting Dialog Message
      alertDialog.setMessage("GPS is not enabled. Do you want to go to
settings menu?");

      // On pressing Settings button
      alertDialog.setPositiveButton("Settings", new
DialogInterface.OnClickListener() {
         public void onClick(DialogInterface dialog,int which) {
            Intent intent = new
Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
            mContext.startActivity(intent);
         }
      });

      // on pressing cancel button
      alertDialog.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
         public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
         }
      });

      // Showing Alert Message
      alertDialog.show();
   }

   @Override
   public void onLocationChanged(Location location) {
   }

   @Override
   public void onProviderDisabled(String provider) {
   }

   @Override
   public void onProviderEnabled(String provider) {
   }

   @Override
   public void onStatusChanged(String provider, int status, Bundle extras) {
   }

   @Override
   public IBinder onBind(Intent arg0) {
      return null;
   }
}
```

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

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

```
   android:orientation = "vertical" >


   <Button
      android:id = "@+id/button"
      android:layout_width = "fill_parent"
      android:layout_height = "wrap_content"
      android:text = "getlocation"/>

</LinearLayout>
```

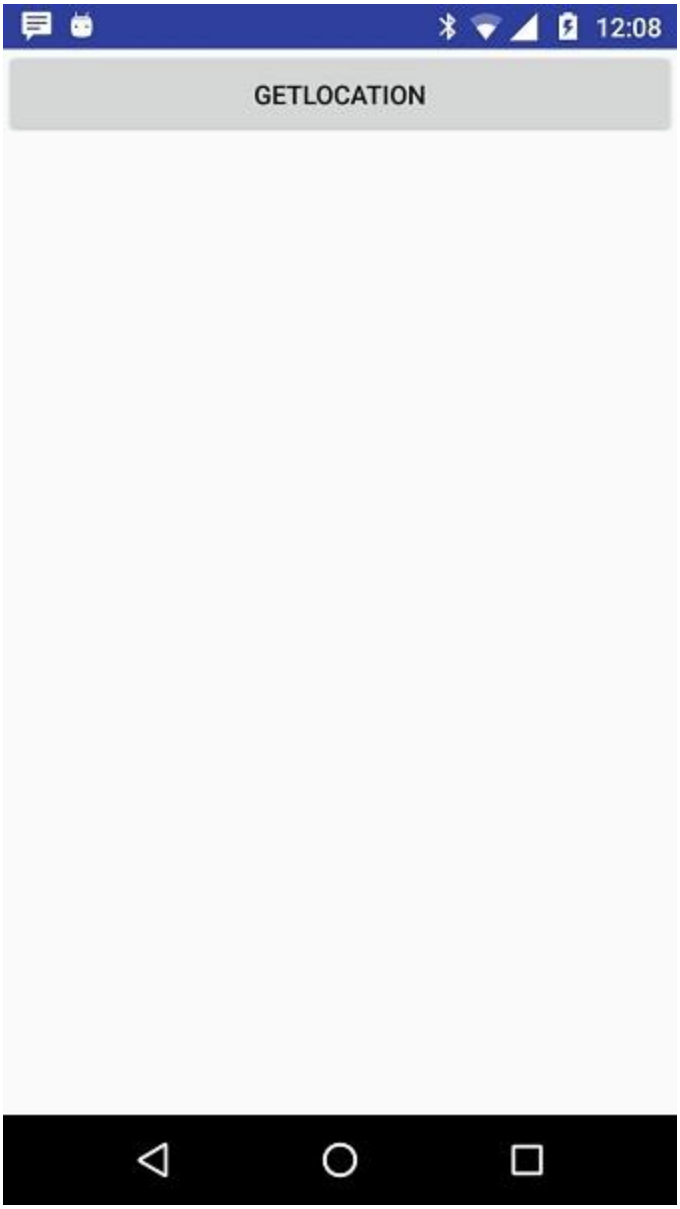
Following will be the content of **res/values/strings.xml** to define two new constants −

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

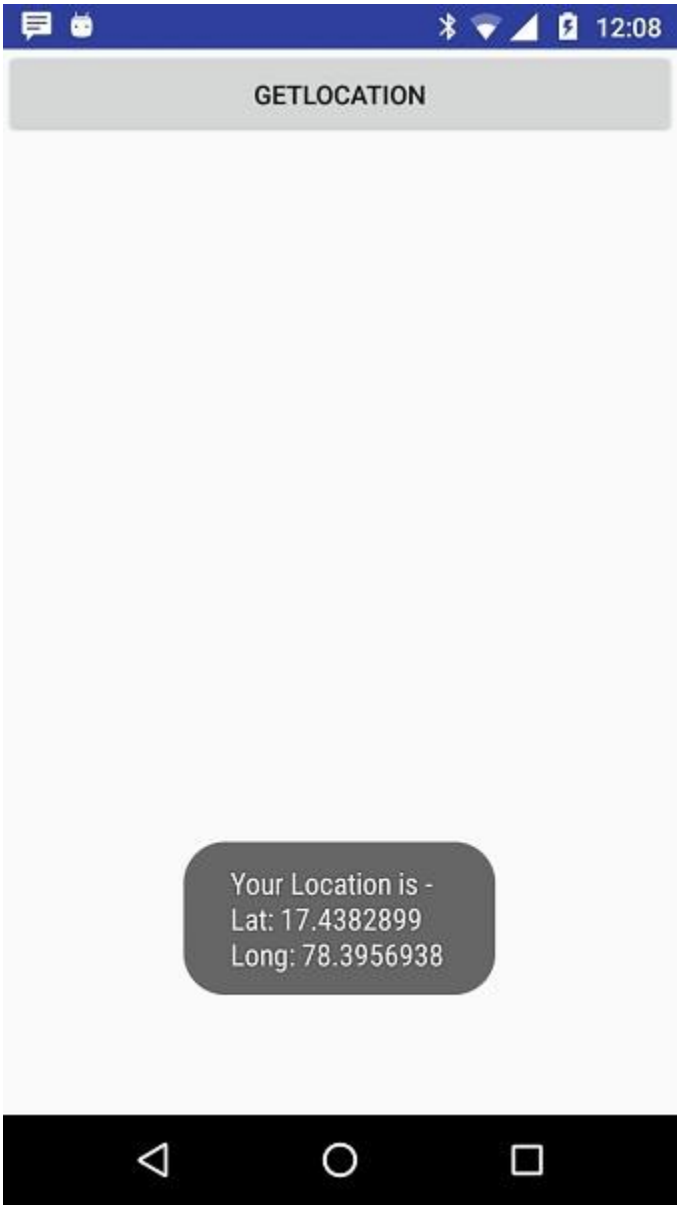
Following is the default content of **AndroidManifest.xml** −

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

I assume that, 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 to see location select Get Location Button which will display location information as follows −


12:08
GETLOCATION
Your Location is -
Lat: 17.4382899
Long: 78.3956938