**GPS- NAVIGATOR**

**Note:**

This tutorial uses older APIs to fetch user’s location. Now google introduced new way of getting user’s location using Google Play Services. Read [Android Location API using Google Play Services](https://www.androidhive.info/2015/02/android-location-api-using-google-play-services/) to get the location using play services.

**Creating new Android Project**

**1**. Create a new project in Eclipse by navigating to **File ⇒ New ⇒ Android Project** and fill all the required details.

**2**. Open AndroidManifest.xml and add **ACCESS\_FINE\_LOCATION** (Which includes both ACCESS\_FINE\_LOCATION and ACCESS\_COARSE\_LOCATION). Also if you are getting network-based location then you need to add**INTERNET** permission too.

|  |
| --- |
| AndroidManifest.xml |
| <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="<http://schemas.android.com/apk/res/android>"      package="com.example.gpstracking"      android:versionCode="1"      android:versionName="1.0" >        <uses-sdk android:minSdkVersion="8" />        <application          android:icon="@drawable/ic\_launcher"          android:label="@string/app\_name" >          <activity              android:name=".AndroidGPSTrackingActivity"              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>        <uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />      <uses-permission android:name="android.permission.INTERNET" />    </manifest> |

**Writing GPS Manager Class**

**3**. Create a new class and name it as **GPSTracker.java** and extend the calls from **Service**. Also implement this class from **LocationListener**.

|  |
| --- |
| public class GPSTracker extends Service implements LocationListener{ |

**4**. Add the required global variables and a constructor for this class.

|  |
| --- |
| 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;        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();      } |

**5**. Add the implementation for the function **geoLocation()** which was called in the constructor. In this function we’ll get the location from network provider first. If network provider is disabled, then we get the location from GPS provider.

|  |
| --- |
| GPSTracker.java |
| 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;      }      @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;      } |

**Getting user’s current location (Latitude and Longitude)**

**6**. Add the following functions to **GPSTracker.java**. (These functions will return 0.00 if it fails to get latitude and longitude)

|  |
| --- |
| GPSTracker.java |
| /\*\*       \* 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;      } |

**Prompting users to Turn On GPS**

**7**. If user turned off the GPS we can use ask user to enable GPS. The following code will show an Alert message asking user to turn on GPS by navigating to GPS Settings automatically.

|  |
| --- |
| GPSTracker.java |
| /\*\*       \* Function to check if best network provider       \* @return boolean       \* \*/      public boolean canGetLocation() {          return this.canGetLocation;      }        /\*\*       \* Function to show settings alert dialog       \* \*/      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?");            // Setting Icon to Dialog          //alertDialog.setIcon(R.drawable.delete);            // 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();      } |

**Stopping the Location Service**

Calling following function will stop using location updates in your application.

|  |
| --- |
| /\*\*       \* Stop using GPS listener       \* Calling this function will stop using GPS in your app       \* \*/      public void stopUsingGPS(){          if(locationManager != null){              locationManager.removeUpdates(GPSTracker.this);          }      } |

**Final Code**

Below is the complete code of GPSTracker.java

|  |
| --- |
| GPSTracker.java |
| package com.example.gpstracking;    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;      }    } |

**How to Use**

**8**. You can get user’s current location by calling simple function from GPSTracker class. Open your main activity and try the following code.

**Check gps enabled or not**

|  |
| --- |
| GPSTracker gps = new GPSTracker(this);  if(gps.canGetLocation()){ // gps enabled} // return boolean true/false |

**Getting Latitude and Longitude**

|  |
| --- |
| gps.getLatitude(); // returns latitude  gps.getLongitude(); // returns longitude |

**Showing GPS Settings Alert Dialog**

|  |
| --- |
| gps.showSettingsAlert(); |

**Stop using GPS**

|  |
| --- |
| gps.stopUsingGPS(); |
| AndroidGPSTrackingActivity.java | |
| package com.example.gpstracking;    import android.app.Activity;  import android.os.Bundle;  import android.view.View;  import android.widget.Button;  import android.widget.Toast;    public class AndroidGPSTrackingActivity extends Activity {        Button btnShowLocation;        // GPSTracker class      GPSTracker gps;        @Override      public void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.main);            btnShowLocation = (Button) findViewById(R.id.btnShowLocation);            // show location button click event          btnShowLocation.setOnClickListener(new View.OnClickListener() {                @Override              public void onClick(View arg0) {                  // create class object                  gps = new GPSTracker(AndroidGPSTrackingActivity.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();                  }                }          });      }    } | |

**Testing your GPS App in Emulator using DDMS Tool**

You can test your application in different ways. If you have real device you can directly test the application by installing your application. If you don’t have one, you can test the app using local emulator.

After starting the emulator open DDMS tool form **EClipse Windows ⇒ Show Perspective ⇒ DDMS** ( Also you can find it on the right corner of IDE)

In the DDMS tool you can find list of emulators you opened. Select the appropriate emulator. In Emulator Controls tab you can manually pass your latitude and longitude to emulator.

You can find Emulator testing in demo video of this tutorial.