Vidyavardhini's College of Engineering & Technology
Department of Computer Engineering
Academic Year: 2023-24

EXPERIMENT NO. 10

AIM: Develop a native application that uses GPS location information.

THEORY:

The Global Positioning System (GPS), originally NAVSTAR GPS, is a satellite-based radio navigation system owned by the United States government and operated by the United States Air Force. It is a global navigation satellite system (GNSS) that provides geo- location and time information to a GPS receiver anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites. Obstacles such as mountains and buildings block the relatively weak GPS signals. The GPS does not require the user to transmit any data, and it operates independently of any telephonic or internet reception, though these technologies can enhance the usefulness of the GPS positioning information. The GPS provides critical positioning capabilities to military, civil, and commercial users around the world. The United States government created the system, maintains it, and makes it freely accessible to anyone with a GPS receiver.

CODE:

```
activity_main.xml
<?xml version = "1.0" encoding = "utf-8"?>
<LinearLayout xmlns:android =
"http://schemas.android.com/apk/res/androi d"
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>
MainActivity.java
package com.example.exp9;
//import
androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import
android.content.pm.PackageManager; import
android.Manifest;
import android.app.Activity; import
android.os.Bundle; 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;
```

```
 overriding
// public void onRequestPermissionsResult(int
requestCode, String[] permissions,
// int[] grantResults)
// to handle the case where the user grants the
permission. See the documentation
// for ActivityCompat#requestPermissions for
more
details. return null;
}
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.NET
W ORK_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,
String mPermission =
Manifest.permission.ACCESS_FINE_LOCATION;
// GPSTracker class GPSTracker
gps;
private PackageManager
MockPackageManager; @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
```

```
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
```

```
// GPS or Network is not enabled
// Ask user to enable GPS/network in settings
gps.showSettingsAlert();
}
}
});
}
}
GPSTracker.java
package com.example.exp9; import
android.Manifest;
import android.app.AlertDialog; import
android.app.Service; import
android.content.Context;
import android.content.DialogInterface; import
android.content.Intent;
import android.content.pm.PackageManager;
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;
import androidx.core.app.ActivityCompat;
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 *
```

```
* @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_SOURC
E_SET TINGS);
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) {
```

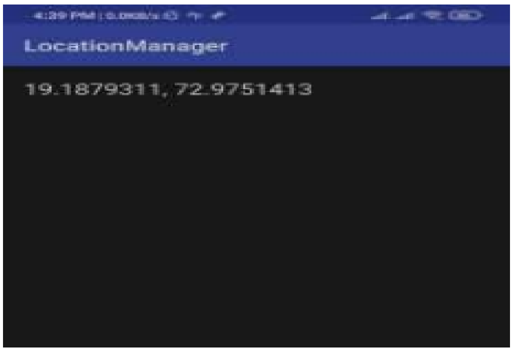
```
60 * 1; // 1
minute
// Declaring a Location Manager
protected LocationManager locationManager;
public
GPSTracker(Context context) { this.mContext =
context;
getLocation();
}
public Location getLocation() { try {
locationManager = (LocationManager)
mContext.getSystemService(LOCATION_SER
VICE
);
// getting GPS status isGPSEnabled
=
locationManager.isProviderEnabled(Location
Manager.GPS_PROVIDER);
// getting network status isNetworkEnabled =
locationManager
.isProviderEnabled(LocationManager.NETWOR
K
_PROVIDER);
if (!isGPSEnabled && !isNetworkEnabled) {
// no network provider is enabled
} else { this.canGetLocation = true;
// First get location from Network Provider
if (isNetworkEnabled) { if
(ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATI
ON)
!= PackageManager.PERMISSION_GRANTED
&& ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOC
ATION
) !=
PackageManager.PERMISSION_GRANTED) {
// TODO: Consider calling
// ActivityCompat#requestPermissions
// here to request the missing permissions, and
then
```

```
}
@Override
public void onProviderEnabled(String provider)
{
}
@Override
public void onStatusChanged(String provider,
int status,
Bundle extras) {
}
@Override
public IBinder onBind(Intent arg0) { return null;
}
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/
apk/res/android"
package="com.example.exp9">
<uses-permission android:name =
"android.permission.ACCESS_FINE_LOCATI
ON"
/>
<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/Theme.Exp9">
<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>
```

OUTPUT:



LocationManager

19.1879311, 72.9751413