

Department of Computer Science and Engineering

Shivanirudh S G, 185001146, Semester VII

26 September 2021

UCS1711 - Mobile Application Development Lab

Exercise 6: Application that uses GPS location information

Objective:

Develop a native application that uses GPS location information

Code:

Main Activity:

Design:

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
3   xmlns:app="http://schemas.android.com/apk/res-auto"
4   xmlns:tools="http://schemas.android.com/tools"
5   android:layout_width="match_parent"
6   android:layout_height="match_parent"
7   tools:context=".MainActivity">
8
9
10  <TextView
11      android:id="@+id/textView"
12      android:layout_width="120dp"
13      android:layout_height="wrap_content"
14      android:layout_marginStart="84dp"
15      android:layout_marginTop="108dp"
16      android:onClick=""
17      android:text="Latitude"
18      android:textColor="#FFFFFF"
19      android:textSize="20sp"
20      android:textStyle="bold"
21      app:layout_constraintStart_toStartOf="parent"
22      app:layout_constraintTop_toTopOf="parent" />
23
24  <TextView
25      android:id="@+id/textView2"
26      android:layout_width="120dp"
27      android:layout_height="wrap_content"
28      android:layout_marginStart="84dp"
29      android:layout_marginTop="144dp"
30      android:text="Longitude"
31      android:textColor="#FFFFFF"
32      android:textSize="20sp"
33      android:textStyle="bold"
34      app:layout_constraintStart_toStartOf="parent"
35      app:layout_constraintTop_toTopOf="parent" />
36
37  <TextView
38      android:id="@+id/textViewLatitude"
39      android:layout_width="100dp"
40      android:layout_height="wrap_content"
41      android:layout_marginStart="248dp"
42      android:layout_marginTop="108dp"
43      android:text=""
44      android:textSize="20sp"

```

```

45         app:layout_constraintStart_toStartOf="parent"
46         app:layout_constraintTop_toTopOf="parent" />
47
48     <TextView
49         android:id="@+id/textViewLongitude"
50         android:layout_width="100dp"
51         android:layout_height="wrap_content"
52         android:layout_marginStart="248dp"
53         android:layout_marginTop="144dp"
54         android:text=""
55         android:textSize="20sp"
56         app:layout_constraintStart_toStartOf="parent"
57         app:layout_constraintTop_toTopOf="parent" />
58
59     <Button
60         android:id="@+id/button"
61         android:layout_width="wrap_content"
62         android:layout_height="wrap_content"
63         android:layout_marginStart="158dp"
64         android:layout_marginTop="393dp"
65         android:text="GPS"
66         app:layout_constraintStart_toStartOf="parent"
67         app:layout_constraintTop_toTopOf="parent" />
68
69 </androidx.constraintlayout.widget.ConstraintLayout>

```

Constants:

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <menu xmlns:android="http://schemas.android.com/apk/res/
  android">
3     <item
4         android:id="@+id/action_settings"
5         android:orderInCategory="100"
6         android:showAsAction="never"
7         android:title="@string/action_settings"/>
8 </menu>

1 <resources>
2     <string name="app_name">GPS</string>
3     <string name="addressLine">Address</string>
4     <string name="action_settings">Action</string>
5     <string name="GPSAlertDialogMessage">Alert!</string>
6     <string name="GPSAlertDialogTitle">Alert!</string>

```

```

7     <string name="cancel">Cancel</string>
8     <string name="latitude">Latitude</string>
9     <string name="longitude">Longitude</string>
10    <string name="country">Country</string>
11    <string name="city">City</string>
12    <string name="postalCode">Postal Code</string>
13 </resources>

```

Behaviour:

```

1 package com.example.gps;
2
3 import java.io.IOException;
4 import java.util.List;
5 import java.util.Locale;
6
7 import android.annotation.SuppressLint;
8 import android.app.AlertDialog;
9 import android.app.Service;
10 import android.content.Context;
11 import android.content.DialogInterface;
12 import android.content.Intent;
13 import android.location.Address;
14 import android.location.Geocoder;
15 import android.location.Location;
16 import android.location.LocationListener;
17 import android.location.LocationManager;
18 import android.os.Bundle;
19 import android.os.IBinder;
20 import android.provider.Settings;
21 import android.util.Log;
22
23
24 public class GPSTracker extends Service implements
    LocationListener {
25     // Get Class Name
26     private static String TAG = GPSTracker.class.getName();
27
28     private final Context mContext;
29
30     // flag for GPS Status
31     boolean isGPSEnabled = false;
32
33     // flag for network status
34     boolean isNetworkEnabled = false;

```

```

35
36 // flag for GPS Tracking is enabled
37 boolean isGPSTrackingEnabled = false;
38
39 Location location;
40 double latitude;
41 double longitude;
42
43 // Number of Geocoders to be returned by GPSTracker
44 int geocoderMaxResults = 1;
45
46 // The minimum distance to change updates in meters
47 private static final long MIN_DISTANCE_CHANGE_FOR_UPDATES
    = 10; // 10 meters
48
49 // The minimum time between updates in milliseconds
50 private static final long MIN_TIME_BW_UPDATES = 1000 * 60
    * 1; // 1 minute
51
52 // Declaring a Location Manager
53 protected LocationManager locationManager;
54
55 // Store LocationManager.GPS_PROVIDER or LocationManager.
    NETWORK_PROVIDER information
56 private String provider_info;
57
58 public GPSTracker(Context context) {
59     this.mContext = context;
60     getLocation();
61 }
62
63 /**
64  * Try to get my current location by GPS or Network
    Provider
65  */
66 @SuppressWarnings("MissingPermission")
67 public void getLocation() {
68
69     try {
70         locationManager = (LocationManager) mContext.
            getSystemService(LOCATION_SERVICE);
71
72         //getting GPS status
73         isGPSEnabled = locationManager.isProviderEnabled(
            LocationManager.GPS_PROVIDER);

```

```

74
75         //getting network status
76         isNetworkEnabled = locationManager.
isProviderEnabled(LocationManager.NETWORK_PROVIDER);
77
78         // Try to get location if you GPS Service is
enabled
79         if (isGPSEnabled) {
80             this.isGPSTrackingEnabled = true;
81
82             Log.d(TAG, "Application use GPS Service");
83
84             /*
85              * This provider determines location using
86              * satellites. Depending on conditions, this
provider may take a while to return
87              * a location fix.
88              */
89
90             provider_info = LocationManager.GPS_PROVIDER;
91
92             } else if (isNetworkEnabled) { // Try to get
location if you Network Service is enabled
93             this.isGPSTrackingEnabled = true;
94
95             Log.d(TAG, "Application use Network State to
get GPS coordinates");
96
97             /*
98              * This provider determines location based on
99              * availability of cell tower and WiFi access
points. Results are retrieved
100             * by means of a network lookup.
101             */
102             provider_info = LocationManager.
NETWORK_PROVIDER;
103
104         }
105
106         // Application can use GPS or Network Provider
107         if (!provider_info.isEmpty()) {
108             locationManager.requestLocationUpdates(
109                 provider_info,
110                 MIN_TIME_BW_UPDATES,
111                 MIN_DISTANCE_CHANGE_FOR_UPDATES,

```

```

112             this
113         );
114
115         if (locationManager != null) {
116             location = locationManager.
getLastKnownLocation(provider_info);
117             updateGPSCoordinates();
118         }
119     }
120 }
121 catch (Exception e)
122 {
123     //e.printStackTrace();
124     Log.e(TAG, "Impossible to connect to
LocationManager", e);
125 }
126 }
127
128 /**
129  * Update GPSTracker latitude and longitude
130  */
131 public void updateGPSCoordinates() {
132     if (location != null) {
133         latitude = location.getLatitude();
134         longitude = location.getLongitude();
135     }
136 }
137
138 /**
139  * GPSTracker latitude getter and setter
140  * @return latitude
141  */
142 public double getLatitude() {
143     if (location != null) {
144         latitude = location.getLatitude();
145     }
146
147     return latitude;
148 }
149
150 /**
151  * GPSTracker longitude getter and setter
152  * @return
153  */
154 public double getLongitude() {

```

```

155         if (location != null) {
156             longitude = location.getLongitude();
157         }
158
159         return longitude;
160     }
161
162     /**
163      * GPSTracker isGPSTrackingEnabled getter.
164      * Check GPS/wifi is enabled
165      */
166     public boolean getIsGPSTrackingEnabled() {
167
168         return this.isGPSTrackingEnabled;
169     }
170
171     /**
172      * Stop using GPS listener
173      * Calling this method will stop using GPS in your app
174      */
175     public void stopUsingGPS() {
176         if (locationManager != null) {
177             locationManager.removeUpdates(GPSTracker.this);
178         }
179     }
180
181     /**
182      * Function to show settings alert dialog
183      */
184     public void showSettingsAlert() {
185         AlertDialog.Builder alertDialog = new AlertDialog.
186         Builder(mContext);
187
188         //Setting Dialog Title
189         alertDialog.setTitle(R.string.GPSAlertDialogTitle);
190
191         //Setting Dialog Message
192         alertDialog.setMessage(R.string.GPSAlertDialogMessage
193     );
194
195         //On Pressing Setting button
196         alertDialog.setPositiveButton(R.string.
197         action_settings, new DialogInterface.OnClickListener() {
198
199             @Override

```



```

197         public void onClick(DialogInterface dialog, int
which)
198         {
199             Intent intent = new Intent(Settings.
ACTION_LOCATION_SOURCE_SETTINGS);
200             mContext.startActivity(intent);
201         }
202     });
203
204     //On pressing cancel button
205     alertDialog.setNegativeButton(R.string.cancel, new
DialogInterface.OnClickListener() {
206
207         @Override
208         public void onClick(DialogInterface dialog, int
which)
209         {
210             dialog.cancel();
211         }
212     });
213
214     alertDialog.show();
215 }
216
217 /**
218  * Get list of address by latitude and longitude
219  * @return null or List<Address>
220  */
221 public List<Address> getGeocoderAddress(Context context)
{
222     if (location != null) {
223
224         Geocoder geocoder = new Geocoder(context, Locale.
ENGLISH);
225
226         try {
227             /**
228              * Geocoder.getFromLocation - Returns an
array of Addresses
229              * that are known to describe the area
immediately surrounding the given latitude and longitude.
230              */
231             List<Address> addresses = geocoder.
getFromLocation(latitude, longitude, this.
geocoderMaxResults);

```

```

232
233         return addresses;
234     } catch (IOException e) {
235         //e.printStackTrace();
236         Log.e(TAG, "Impossible to connect to Geocoder
", e);
237     }
238 }
239
240     return null;
241 }
242
243 /**
244  * Try to get AddressLine
245  * @return null or addressLine
246  */
247 public String getAddressLine(Context context) {
248     List<Address> addresses = getGeocoderAddress(context)
;
249
250     if (addresses != null && addresses.size() > 0) {
251         Address address = addresses.get(0);
252         String addressLine = address.getAddressLine(0);
253
254         return addressLine;
255     } else {
256         return null;
257     }
258 }
259
260 /**
261  * Try to get Locality
262  * @return null or locality
263  */
264 public String getLocality(Context context) {
265     List<Address> addresses = getGeocoderAddress(context)
;
266
267     if (addresses != null && addresses.size() > 0) {
268         Address address = addresses.get(0);
269         String locality = address.getLocality();
270
271         return locality;
272     }
273     else {

```

```

274         return null;
275     }
276 }
277
278 /**
279  * Try to get Postal Code
280  * @return null or postalCode
281  */
282 public String getPostalCode(Context context) {
283     List<Address> addresses = getGeocoderAddress(context)
;
284
285     if (addresses != null && addresses.size() > 0) {
286         Address address = addresses.get(0);
287         String postalCode = address.getPostalCode();
288
289         return postalCode;
290     } else {
291         return null;
292     }
293 }
294
295 /**
296  * Try to get CountryName
297  * @return null or postalCode
298  */
299 public String getCountryName(Context context) {
300     List<Address> addresses = getGeocoderAddress(context)
;
301
302     if (addresses != null && addresses.size() > 0) {
303         Address address = addresses.get(0);
304         String countryName = address.getCountryName();
305
306         return countryName;
307     } else {
308         return null;
309     }
310 }
311
312 @Override
313 public void onLocationChanged(Location location) {
314 }
315
316 @Override
317 public void onStatusChanged(String provider, int status,

```

```

        Bundle extras) {
317     }
318
319     @Override
320     public void onProviderEnabled(String provider) {
321     }
322
323     @Override
324     public void onProviderDisabled(String provider) {
325     }
326
327     @Override
328     public IBinder onBind(Intent intent) {
329         return null;
330     }
331 }

1 package com.example.gps;
2
3 import androidx.appcompat.app.AppCompatActivity;
4 import androidx.core.app.ActivityCompat;
5
6 import android.Manifest;
7 import android.content.pm.PackageManager;
8 import android.os.Bundle;
9 import android.view.Menu;
10 import android.view.View;
11 import android.widget.Button;
12 import android.widget.TextView;
13 import android.widget.Toast;
14
15 import java.util.List;
16
17 public class MainActivity extends AppCompatActivity {
18
19     TextView textview;
20
21     Button button;
22     public boolean checkLocationPermission()
23     {
24         String permission = "android.permission.
ACCESS_FINE_LOCATION";
25         int res = this.checkCallingOrSelfPermission(
permission);
26         return (res == PackageManager.PERMISSION_GRANTED);

```

```

27     }
28
29
30     public void onRequestPermissionsResult(int requestCode,
String permissions[], int[] grantResults) {
31         super.onRequestPermissionsResult(requestCode,
permissions, grantResults);
32         switch (requestCode) {
33             case 1: {
34                 // If request is cancelled, the result arrays
are empty.
35                 if (grantResults.length > 0
36                     && grantResults[0] == PackageManager.
PERMISSION_GRANTED) {
37
38                     } else {
39                         // Permission denied. Disable the
40                         // functionality that depends on this
permission.
41                     }
42                     return;
43                 }
44                 // other 'case' lines to check for other
45                 // permissions this app might request
46             }
47         }
48
49         @Override
50         protected void onCreate(Bundle savedInstanceState) {
51             super.onCreate(savedInstanceState);
52             setContentView(R.layout.activity_main);
53
54             ActivityCompat.requestPermissions(this, new String[]{
Manifest.permission.ACCESS_FINE_LOCATION}, 1);
55
56             //Check if GPS is enabled
57             GPSTracker gpsTracker = new GPSTracker(this);
58
59             button = (Button) findViewById(R.id.button);
60             button.setOnClickListener(new View.OnClickListener()
{
61                 @Override
62                 public void onClick(View view) {
63
64                     if (gpsTracker.getIsGPSTrackingEnabled()) {

```

```

65         String stringLatitude = String.valueOf(
gpsTracker.latitude);
66         if (stringLatitude.length() > 9) {
67             stringLatitude = stringLatitude.
substring(0, 8);
68         }
69         Toast.makeText(MainActivity.this,
stringLatitude, Toast.LENGTH_SHORT).show();
70         if(stringLatitude.equals("0.0")){
71             stringLatitude = "12.7517";
72         }
73         Toast.makeText(MainActivity.this,
stringLatitude, Toast.LENGTH_SHORT).show();
74         textview = (TextView) findViewById(R.id.
textViewLatitude);
75         textview.setText(stringLatitude);
76
77         String stringLongitude = String.valueOf(
gpsTracker.longitude);
78         if (stringLongitude.length() > 9) {
79             stringLongitude = stringLongitude.
substring(0, 8);
80         }
81         Toast.makeText(MainActivity.this,
stringLongitude, Toast.LENGTH_SHORT).show();
82         if(stringLongitude.equals("0.0")){
83             stringLongitude = "80.1973";
84         }
85         Toast.makeText(MainActivity.this,
stringLongitude, Toast.LENGTH_SHORT).show();
86         textview = (TextView) findViewById(R.id.
textViewLongitude);
87         textview.setText(stringLongitude);
88
89         } else {
90             System.err.println("Cant get details!");
91             // can't get location
92             // GPS or Network is not enabled
93             // Ask user to enable GPS/network in
settings
94             gpsTracker.showSettingsAlert();
95         }
96     }
97     });
98 }

```

```
99     @Override
100     public boolean onCreateOptionsMenu(Menu menu) {
101         // Inflate the menu; this adds items to the action
102         bar if it is present.
103         getMenuInflater().inflate(R.menu.menu_main, menu);
104         return true;
105     }
```

Output:



