**UCS6611 - MOBILE APPLICATION DEVELOPMENT LABORATORY**

**Ex. No. 6 Develop a native application that uses GPS location information**

**Date:** 30/9/21 **Name:** Srinath S

**Class:** CSE-C  **Roll:** 185001205

**Question:**

Develop a native application that uses GPS location information

**Code:**

**Activitymain.xml:**

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

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:app="http://schemas.android.com/apk/res-auto"

    xmlns:tools="http://schemas.android.com/tools"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    tools:context=".MainActivity">

    xmlns:tools="https://schemas.android.com/tools"

        android:id="@+id/activity\_main"

        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="com.journaldev.gpslocationtracking.MainActivity">

        <Button

            android:layout\_width="wrap\_content"

            android:layout\_height="wrap\_content"

            android:id="@+id/btn"

            android:layout\_centerInParent="true"

            android:text="GET LOCATION" />

</androidx.constraintlayout.widget.ConstraintLayout>

**Mainactivity.java:**

package com.example.geolocation;

import android.annotation.TargetApi;

import android.app.AlertDialog;

import android.content.DialogInterface;

import android.content.pm.PackageManager;

import android.os.Build;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.util.ArrayList;

import static android.Manifest.permission.ACCESS\_COARSE\_LOCATION;

import static android.Manifest.permission.ACCESS\_FINE\_LOCATION;

public class MainActivity extends AppCompatActivity {

    private ArrayList permissionsToRequest;

    private ArrayList permissionsRejected = new ArrayList();

    private ArrayList permissions = new ArrayList();

    private final static int ALL\_PERMISSIONS\_RESULT = 101;

    LocationTrack locationTrack;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        permissions.add(ACCESS\_FINE\_LOCATION);

        permissions.add(ACCESS\_COARSE\_LOCATION);

        permissionsToRequest = findUnAskedPermissions(permissions);

        if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.M) {

            if (permissionsToRequest.size() > 0)

                requestPermissions((String[]) permissionsToRequest.toArray(new String[permissionsToRequest.size()]), ALL\_PERMISSIONS\_RESULT);

        }

        Button btn = (Button) findViewById(R.id.btn);

        btn.setOnClickListener(new View.OnClickListener() {

            @Override

            public void onClick(View view) {

                locationTrack = new LocationTrack(MainActivity.this);

                if (locationTrack.canGetLocation()) {

                    double longitude = locationTrack.getLongitude();

                    double latitude = locationTrack.getLatitude();

                    Toast.makeText(getApplicationContext(), "Longitude:" + Double.toString(longitude) + "\nLatitude:" + Double.toString(latitude), Toast.LENGTH\_SHORT).show();

                } else {

                    locationTrack.showSettingsAlert();

                }

            }

        });

    }

    private ArrayList findUnAskedPermissions(ArrayList wanted) {

        ArrayList result = new ArrayList();

        for (Object perm : wanted) {

            if (!hasPermission((String) perm)) {

                result.add(perm);

            }

        }

        return result;

    }

    private boolean hasPermission(String permission) {

        if (canMakeSmores()) {

            if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.M) {

                return (checkSelfPermission(permission) == PackageManager.PERMISSION\_GRANTED);

            }

        }

        return true;

    }

    private boolean canMakeSmores() {

        return (Build.VERSION.SDK\_INT > Build.VERSION\_CODES.LOLLIPOP\_MR1);

    }

    @TargetApi(Build.VERSION\_CODES.M)

    @Override

    public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {

        switch (requestCode) {

            case ALL\_PERMISSIONS\_RESULT:

                for (Object perms : permissionsToRequest) {

                    if (!hasPermission((String) perms)) {

                        permissionsRejected.add(perms);

                    }

                }

                if (permissionsRejected.size() > 0) {

                    if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.M) {

                        if (shouldShowRequestPermissionRationale((String) permissionsRejected.get(0))) {

                            showMessageOKCancel("These permissions are mandatory for the application. Please allow access.",

                                    new DialogInterface.OnClickListener() {

                                        @Override

                                        public void onClick(DialogInterface dialog, int which) {

                                            if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.M) {

                                                requestPermissions((String[]) permissionsRejected.toArray(new String[permissionsRejected.size()]), ALL\_PERMISSIONS\_RESULT);

                                            }

                                        }

                                    });

                            return;

                        }

                    }

                }

                break;

        }

    }

    private void showMessageOKCancel(String message, DialogInterface.OnClickListener okListener) {

        new AlertDialog.Builder(MainActivity.this)

                .setMessage(message)

                .setPositiveButton("OK", okListener)

                .setNegativeButton("Cancel", null)

                .create()

                .show();

    }

    @Override

    protected void onDestroy() {

        super.onDestroy();

        locationTrack.stopListener();

    }

}

**Location.java:**

package com.example.geolocation;

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.widget.Toast;

import androidx.core.app.ActivityCompat;

public class LocationTrack extends Service implements LocationListener {

    private final Context mContext;

    boolean checkGPS = false;

    boolean checkNetwork = false;

    boolean canGetLocation = false;

    Location loc;

    double latitude;

    double longitude;

    private static final long MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES = 10;

    private static final long MIN\_TIME\_BW\_UPDATES = 1000 \* 60 \* 1;

    protected LocationManager locationManager;

    public LocationTrack(Context mContext) {

        this.mContext = mContext;

        getLocation();

    }

    private Location getLocation() {

        try {

            locationManager = (LocationManager) mContext

                    .getSystemService(LOCATION\_SERVICE);

            checkGPS = locationManager

                    .isProviderEnabled(LocationManager.GPS\_PROVIDER);

            checkNetwork = locationManager

                    .isProviderEnabled(LocationManager.NETWORK\_PROVIDER);

            if (!checkGPS && !checkNetwork) {

                Toast.makeText(mContext, "No Service Provider is available", Toast.LENGTH\_SHORT).show();

            } else {

                this.canGetLocation = true;

                if (checkGPS) {

                    if (ActivityCompat.checkSelfPermission(mContext, Manifest.permission.ACCESS\_FINE\_LOCATION) != PackageManager.PERMISSION\_GRANTED && ActivityCompat.checkSelfPermission(mContext, Manifest.permission.ACCESS\_COARSE\_LOCATION) != PackageManager.PERMISSION\_GRANTED) {

                    }

                    locationManager.requestLocationUpdates(

                            LocationManager.GPS\_PROVIDER,

                            MIN\_TIME\_BW\_UPDATES,

                            MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES, this);

                    if (locationManager != null) {

                        loc = locationManager

                                .getLastKnownLocation(LocationManager.GPS\_PROVIDER);

                        if (loc != null) {

                            latitude = loc.getLatitude();

                            longitude = loc.getLongitude();

                        }

                    }

                }

            }

        } catch (Exception e) {

            e.printStackTrace();

        }

        return loc;

    }

    public double getLongitude() {

        if (loc != null) {

            longitude = loc.getLongitude();

        }

        return longitude;

    }

    public double getLatitude() {

        if (loc != null) {

            latitude = loc.getLatitude();

        }

        return latitude;

    }

    public boolean canGetLocation() {

        return this.canGetLocation;

    }

    public void showSettingsAlert() {

        AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext);

        alertDialog.setTitle("GPS is not Enabled!");

        alertDialog.setMessage("Do you want to turn on GPS?");

        alertDialog.setPositiveButton("Yes", new DialogInterface.OnClickListener() {

            public void onClick(DialogInterface dialog, int which) {

                Intent intent = new Intent(Settings.ACTION\_LOCATION\_SOURCE\_SETTINGS);

                mContext.startActivity(intent);

            }

        });

        alertDialog.setNegativeButton("No", new DialogInterface.OnClickListener() {

            public void onClick(DialogInterface dialog, int which) {

                dialog.cancel();

            }

        });

        alertDialog.show();

    }

    public void stopListener() {

        if (locationManager != null) {

            if (ActivityCompat.checkSelfPermission(mContext, Manifest.permission.ACCESS\_FINE\_LOCATION) != PackageManager.PERMISSION\_GRANTED && ActivityCompat.checkSelfPermission(mContext, Manifest.permission.ACCESS\_COARSE\_LOCATION) != PackageManager.PERMISSION\_GRANTED) {

                return;

            }

            locationManager.removeUpdates(LocationTrack.this);

        }

    }

    @Override

    public IBinder onBind(Intent intent) {

        return null;

    }

    @Override

    public void onLocationChanged(Location location) {

    }

    @Override

    public void onStatusChanged(String s, int i, Bundle bundle) {

    }

    @Override

    public void onProviderEnabled(String s) {

    }

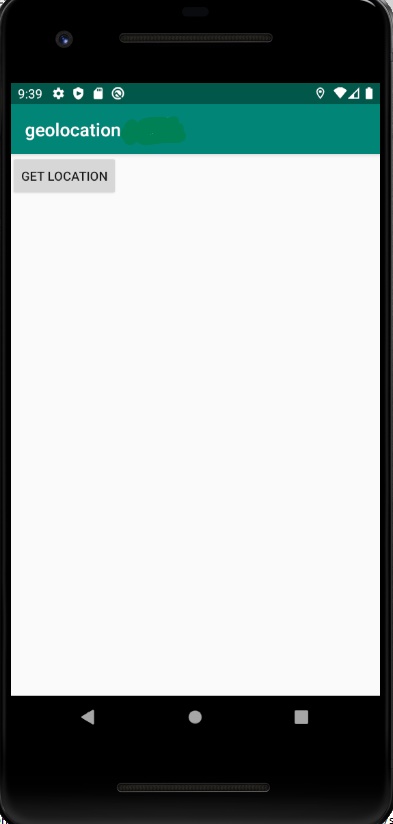
    @Override

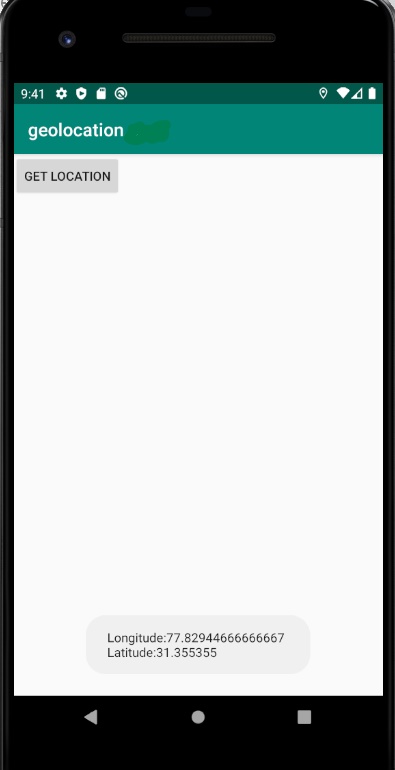
    public void onProviderDisabled(String s) {

    }

}

**Output:**

** **

****