Code:

public void onLocationChanged(Location location) {

Log.d("test", "onLocationChanged Called");

double latitude = location.getLatitude();

double longitude = location.getLongitude();

LatLng latLng = new LatLng(latitude, longitude);

gMap.addMarker(new MarkerOptions().position(latLng));

gMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));

gMap.animateCamera(CameraUpdateFactory.zoomTo(15));

Log.d("location", "Latitude:" + latitude + ", Longitude:" + longitude);

}

Permission to manifest: <uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION"/>

Java coding:

import android.Manifest;

import android.content.pm.PackageManager;

import android.graphics.Point;

import android.location.Location;

import android.os.Bundle;

import android.os.Handler;

import android.os.Message;

import android.os.SystemClock;

import android.support.annotation.NonNull;

import android.support.annotation.Nullable;

import android.support.v4.app.ActivityCompat;

import android.support.v4.app.FragmentActivity;

import android.support.v4.content.ContextCompat;

import android.util.Log;

import android.view.animation.Interpolator;

import android.view.animation.LinearInterpolator;

import android.widget.Button;

import android.widget.TextView;

import com.google.android.gms.common.ConnectionResult;

import com.google.android.gms.common.api.GoogleApiClient;

import com.google.android.gms.common.api.PendingResult;

import com.google.android.gms.common.api.Status;

import com.google.android.gms.location.LocationListener;

import com.google.android.gms.location.LocationRequest;

import com.google.android.gms.location.LocationServices;

import com.google.android.gms.maps.CameraUpdateFactory;

import com.google.android.gms.maps.GoogleMap;

import com.google.android.gms.maps.OnMapReadyCallback;

import com.google.android.gms.maps.Projection;

import com.google.android.gms.maps.SupportMapFragment;

import com.google.android.gms.maps.model.LatLng;

import com.google.android.gms.maps.model.Marker;

import com.google.android.gms.maps.model.MarkerOptions;

public class MapsActivity extends FragmentActivity implements LocationListener,

OnMapReadyCallback, GoogleApiClient

.ConnectionCallbacks, GoogleApiClient.OnConnectionFailedListener {

private GoogleMap mMap;

private final int MY\_LOCATION\_REQUEST\_CODE = 100;

private Handler handler;

private Marker m;

// private GoogleApiClient googleApiClient;

public final static int SENDING = 1;

public final static int CONNECTING = 2;

public final static int ERROR = 3;

public final static int SENT = 4;

public final static int SHUTDOWN = 5;

private static final String TAG = "LocationActivity";

private static final long INTERVAL = 1000 \* 10;

private static final long FASTEST\_INTERVAL = 1000 \* 5;

Button btnFusedLocation;

TextView tvLocation;

LocationRequest mLocationRequest;

GoogleApiClient mGoogleApiClient;

Location mCurrentLocation;

String mLastUpdateTime;

private Location previousLocation;

protected void createLocationRequest() {

mLocationRequest = new LocationRequest();

mLocationRequest.setInterval(INTERVAL);

mLocationRequest.setFastestInterval(FASTEST\_INTERVAL);

mLocationRequest.setPriority(LocationRequest.PRIORITY\_HIGH\_ACCURACY);

}

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_maps);

createLocationRequest();

mGoogleApiClient = new GoogleApiClient.Builder(this)

.addApi(LocationServices.API)

.addConnectionCallbacks(this)

.addOnConnectionFailedListener(this)

.build();

// Obtain the SupportMapFragment and get notified when the map is ready to be used.

SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()

.findFragmentById(R.id.map);

mapFragment.getMapAsync(this);

handler = new Handler() {

@Override

public void handleMessage(Message msg) {

switch (msg.what) {

case SENDING:

break;

}

}

};

}

/\*\*

\* Manipulates the map once available.

\* This callback is triggered when the map is ready to be used.

\* This is where we can add markers or lines, add listeners or move the camera. In this case,

\* we just add a marker near Sydney, Australia.

\* If Google Play services is not installed on the device, the user will be prompted to install

\* it inside the SupportMapFragment. This method will only be triggered once the user has

\* installed Google Play services and returned to the app.

\*/

@Override

public void onMapReady(GoogleMap googleMap) {

mMap = googleMap;

// Add a marker in Sydney and move the camera

LatLng sydney = new LatLng(-34, 151);

m = mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in " +

"Sydney"));

mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));

if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS\_FINE\_LOCATION)

== PackageManager.PERMISSION\_GRANTED) {

mMap.setMyLocationEnabled(true);

} else {

// Show rationale and request permission.

}

}

public void rotateMarker(final Marker marker, final float toRotation, final float st) {

final Handler handler = new Handler();

final long start = SystemClock.uptimeMillis();

final float startRotation = st;

final long duration = 1555;

final Interpolator interpolator = new LinearInterpolator();

handler.post(new Runnable() {

@Override

public void run() {

long elapsed = SystemClock.uptimeMillis() - start;

float t = interpolator.getInterpolation((float) elapsed / duration);

float rot = t \* toRotation + (1 - t) \* startRotation;

marker.setRotation(-rot > 180 ? rot / 2 : rot);

if (t < 1.0) {

// Post again 16ms later.

handler.postDelayed(this, 16);

}

}

});

}

public void animateMarker(final LatLng toPosition, final boolean hideMarke) {

final Handler handler = new Handler();

final long start = SystemClock.uptimeMillis();

Projection proj = mMap.getProjection();

Point startPoint = proj.toScreenLocation(m.getPosition());

final LatLng startLatLng = proj.fromScreenLocation(startPoint);

final long duration = 5000;

final Interpolator interpolator = new LinearInterpolator();

handler.post(new Runnable() {

@Override

public void run() {

long elapsed = SystemClock.uptimeMillis() - start;

float t = interpolator.getInterpolation((float) elapsed

/ duration);

double lng = t \* toPosition.longitude + (1 - t)

\* startLatLng.longitude;

double lat = t \* toPosition.latitude + (1 - t)

\* startLatLng.latitude;

m.setPosition(new LatLng(lat, lng));

if (t < 1.0) {

// Post again 16ms later.

handler.postDelayed(this, 16);

} else {

if (hideMarke) {

m.setVisible(false);

} else {

m.setVisible(true);

}

}

}

});

}

private double bearingBetweenLocations(LatLng latLng1, LatLng latLng2) {

double PI = 3.14159;

double lat1 = latLng1.latitude \* PI / 180;

double long1 = latLng1.longitude \* PI / 180;

double lat2 = latLng2.latitude \* PI / 180;

double long2 = latLng2.longitude \* PI / 180;

double dLon = (long2 - long1);

double y = Math.sin(dLon) \* Math.cos(lat2);

double x = Math.cos(lat1) \* Math.sin(lat2) - Math.sin(lat1)

\* Math.cos(lat2) \* Math.cos(dLon);

double brng = Math.atan2(y, x);

brng = Math.toDegrees(brng);

brng = (brng + 360) % 360;

return brng;

}

/\* @Override

public void onRequestPermissionsResult(int requestCode, String[] permissions, int[]

grantResults) {

if (requestCode == MY\_LOCATION\_REQUEST\_CODE) {

if (permissions.length == 1 &&

permissions[0] == Manifest.permission.ACCESS\_FINE\_LOCATION &&

grantResults[0] == PackageManager.PERMISSION\_GRANTED) {

mMap.setMyLocationEnabled(true);

} else {

// Permission was denied. Display an error message.

}

}

}\*/

@Override

public void onConnected(@Nullable Bundle bundle) {

Log.d(TAG, "onConnected - isConnected ...............: " + mGoogleApiClient.isConnected());

startLocationUpdates();

}

@Override

protected void onStart() {

super.onStart();

Log.d(TAG, "onStart fired ..............");

mGoogleApiClient.connect();

}

@Override

protected void onStop() {

super.onStop();

mGoogleApiClient.disconnect();

Log.d(TAG, "isConnected ...............: " + mGoogleApiClient.isConnected());

}

@Override

public void onConnectionSuspended(int i) {

}

@Override

public void onConnectionFailed(@NonNull ConnectionResult connectionResult) {

}

protected void startLocationUpdates() {

if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS\_FINE\_LOCATION) !=

PackageManager.PERMISSION\_GRANTED && ActivityCompat.checkSelfPermission(this,

Manifest.permission.ACCESS\_COARSE\_LOCATION) != PackageManager.PERMISSION\_GRANTED) {

// TODO: Consider calling

// ActivityCompat#requestPermissions

// here to request the missing permissions, and then 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;

}

PendingResult<Status> pendingResult = LocationServices.FusedLocationApi

.requestLocationUpdates(mGoogleApiClient, mLocationRequest, this);

Log.d(TAG, "Location update started ..............: ");

}

LatLng previouslatLng;

@Override

public void onLocationChanged(Location location) {

previouslatLng = new LatLng(location.getLatitude(), location.getLongitude());

double rota = 0.0;

double startrota = 0.0;

if (previousLocation != null) {

rota = bearingBetweenLocations(previouslatLng, new LatLng(location.getLatitude

(), location.getLongitude()));

}

rotateMarker(m, (float) rota, (float) startrota);

previousLocation = location;

Log.d(TAG, "Firing onLocationChanged..........................");

Log.d(TAG, "lat :" + location.getLatitude() + "long :" + location.getLongitude());

Log.d(TAG, "bearing :" + location.getBearing());

animateMarker(new LatLng(location.getLatitude(), location.getLongitude()), false);

// new ServerConnAsync(handler, MapsActivity.this,location).execute();

}

@Override

protected void onPause() {

super.onPause();

stopLocationUpdates();

}

protected void stopLocationUpdates() {

LocationServices.FusedLocationApi.removeLocationUpdates(

mGoogleApiClient, this);

Log.d(TAG, "Location update stopped .......................");

}

@Override

public void onResume() {

super.onResume();

if (mGoogleApiClient.isConnected()) {

startLocationUpdates();

Log.d(TAG, "Location update resumed .....................");

}

}

}

Xml activity:

<fragment android:id="@+id/map"

android:name="com.google.android.gms.maps.SupportMapFragment"

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

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"