

# CSE 162 Mobile Computing

## Lab 3 Location and Map Programming

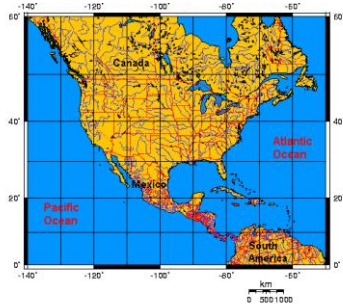
Hua Huang

Lecture: Feb 16 / Feb 18

Demo: Feb 23 / Feb 25

Hard Deadline: Mar 2 / Mar 4

# What is localization, aka location?



absolute location  
(lat, long)



"I hope this bullhorn will make this meeting a little less boring."

relative location



context location

# Why should I care about localization?



social application



advertisements



podcasting

# Assignments

- Creating a google map, display the map
- Using Location services, set the map view based on the GPS tracking

# Create Project

- Create new project: MapsProject
- Select google map activity
- Go to: google\_maps\_api.xml

</resources>

<!--

*TODO: Before you run your application, you need a Google Maps API key.*

To get one, follow this link, follow the directions and press "Create" at the end:

[https://console.developers.google.com/flows/enableapi?apiid=maps\\_android\\_backend&keyType=CLIENT\\_SIDE](https://console.developers.google.com/flows/enableapi?apiid=maps_android_backend&keyType=CLIENT_SIDE)

You can also add your credentials to an existing key, using these values:

Package name:

com.example.mapsproject

SHA-1 certificate fingerprint:

5B:E5:C7:B2:82:C2:84:A0:77:AE:09:F6:69:5B:14:47:BA:68:A0:C4

Alternatively, follow the directions here:

<https://developers.google.com/maps/documentation/android/start#get-key>

Once you have your key (it starts with "AIza"), replace the "google\_maps\_key" string in this file.

-->


<string name="google\_maps\_key" templateMergeStrategy="preserve" translatable="false">YOUR\_KEY\_HERE<

</resources>

## Credentials







[+ CREATE CREDENTIALS](#) DELETE

Create credentials to access your enabled APIs. [Learn more](#)

 Remember to configure the OAuth consent screen with information about your application.

[CONFIGURE CONSENT SCREEN](#)

## API Keys

<input type="checkbox"/>	Name	Creation date ↓	Restrictions	Key			
<input type="checkbox"/>	✓ API key 1	Feb 8, 2021	Android apps	AIzaSyBGMJ...fGCmPhxWFk			

</resources>

<!--

*TODO: Before you run your application, you need a Google Maps API key.*

To get one, follow this link, follow the directions and press "Create" at the end:

[https://console.developers.google.com/flows/enableapi?apiid=maps\\_android\\_backend&keyType=CLIENT\\_SIDE](https://console.developers.google.com/flows/enableapi?apiid=maps_android_backend&keyType=CLIENT_SIDE)

You can also add your credentials to an existing key, using these values:

Package name:

com.example.mapsproject

SHA-1 certificate fingerprint:

5B:E5:C7:B2:82:C2:84:A0:77:AE:09:F6:69:5B:14:47:BA:68:A0:C4

Alternatively, follow the directions here:

<https://developers.google.com/maps/documentation/android/start#get-key>

Once you have your key (it starts with "AIza"), replace the "google\_maps\_key" string in this file.

-->

<string name="google\_maps\_key" templateMergeStrategy="preserve" translatable="false">YOUR\_KEY\_HERE</string>

</resources>



# Registering for Location Updates

- In order for an object to receive updates from GPS, it must implement the **LocationListener** interface

```
// Called when your GPS location changes
```

```
@Override
```

```
public void onLocationChanged(Location location)
```

```
// Called when a provider gets turned off by the user in the settings
```

```
@Override
```

```
public void onProviderDisabled(String provider)
```

```
// Called when a provider is turned on by the user in the settings
```

```
@Override
```

```
public void onProviderEnabled(String provider)
```

```
// Signals a state change in the GPS (e.g. you head through a tunnel and
```

```
// it loses its fix on your position)
```

```
@Override
```

```
public void onStatusChanged(String provider, int status, Bundle extras)
```

# Registering for Location Updates

- **LocationManager** handles registrations for GPS and network location updates
- Once the **LocationManager** is obtained, an object registers for updates by calling **requestLocationUpdates** , the arguments passed into the requestLocationUpdates method determine the granularity of location changes that will generate an event
  - send updates that are at least X meters apart
  - send updates at least this far apart in time
  - send updates that have this minimum accuracy

**requestLocationUpdates**(String provider, long minTimeMs, float minDistanceM, LocationListener listener)

Register for location updates from the given provider with the given arguments.

**requestLocationUpdates**(long minTimeMs, float minDistanceM, Criteria criteria, LocationListener listener, Looper looper)

Register for location updates using a provider selected through the given Criteria, and a callback on the specified Looper.

**requestLocationUpdates**(String provider, long minTimeMs, float minDistanceM, LocationListener listener, Looper looper)

Register for location updates using the named provider, and a callback on the specified Looper.

**requestLocationUpdates**(String provider, long minTimeMs, float minDistanceM, Executor executor, LocationListener listener)

Register for location updates using the named provider, and a callback on the specified Executor.

**requestLocationUpdates**(long minTimeMs, float minDistanceM, Criteria criteria, PendingIntent pendingIntent)

Register for location updates using a provider selected through the given Criteria, and callbacks delivered via the provided PendingIntent.

**requestLocationUpdates**(long minTimeMs, float minDistanceM, Criteria criteria, Executor executor, LocationListener listener)

Register for location updates using a provider selected through the given Criteria, and a callback on the specified Executor.

**requestLocationUpdates**(String provider, long minTimeMs, float minDistanceM, PendingIntent pendingIntent)

Register for location updates using the named provider, and callbacks delivered via the provided PendingIntent.

# Location Updates

```
public class MapsActivity extends FragmentActivity implements OnMapReadyCallback, LocationListener {
    private GoogleMap mMap;
    private LocationManager locationManager;
    private double longitude = 151;
    private double latitude = -34;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);

        // 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);
        locationManager = (LocationManager) getSystemService(LOCATION_SERVICE);
        if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION) ==
        PackageManager.PERMISSION_GRANTED)
            locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 10,
            Criteria.ACCURACY_COARSE, this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
    }
}
```

# Location Providers

- The phone's location can be determined from multiple providers
- GPS location updates consume significantly more power than network location updates but are more accurate
  - GPS: 25 seconds \* 140mA = 1mAh
  - Network: 2 seconds \* 180mA = 0.1mAh
- The provider argument determines which method will be used to get a location
- You can also register for the PASSIVE\_PROVIDER which only updates you if another app is actively using GPS / Network location

String	<b>GPS_PROVIDER</b> Name of the GPS location provider.
String	<b>NETWORK_PROVIDER</b> Name of the network location provider.
String	<b>PASSIVE_PROVIDER</b> A special location provider for receiving locations without actually initiating a location fix.

# Getting Location Info

```
@Override
public void onLocationChanged(Location location) {
    longitude = location.getLongitude();
    latitude = location.getLatitude();
    AlertDialog alertDialog = new AlertDialog.Builder(MapsActivity.this).create();
    alertDialog.setTitle("Location Detected");
    alertDialog.setMessage(String.valueOf(latitude) + "," + String.valueOf(longitude));
    alertDialog.setButton(AlertDialog.BUTTON_NEUTRAL, "OK",
        new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {
                dialog.dismiss();
            }
        });
    alertDialog.show();
    LatLng loc = new LatLng(latitude, longitude);
    mMap.addMarker(new MarkerOptions().position(loc).title("Marker in at your location"));
    mMap.moveCamera(CameraUpdateFactory.newLatLng(loc));
    mMap.animateCamera(CameraUpdateFactory.newLatLngZoom(loc, 12.0f));
}
```

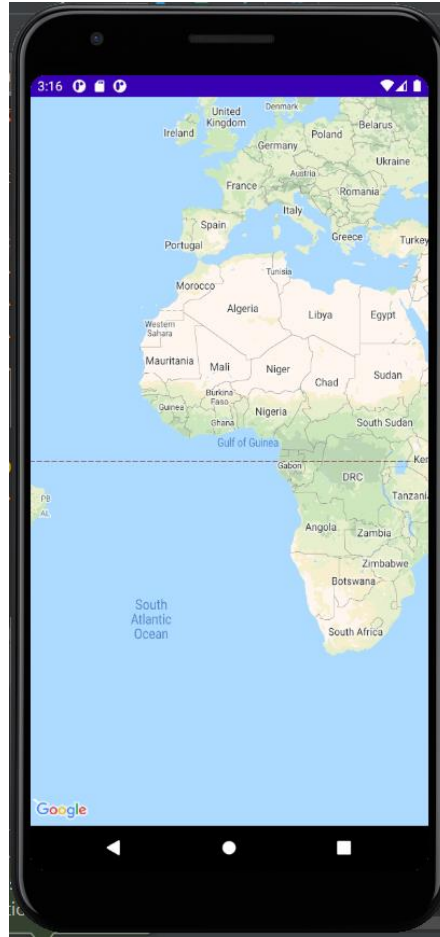
# Being a Good Citizen

- It is important that you unregister your App when you no longer need updates
- For example, you should always unregister your listener when your Activity is paused!
- If you unregister when you pause, you must also reregister when you resume
- This is true for all sensors!

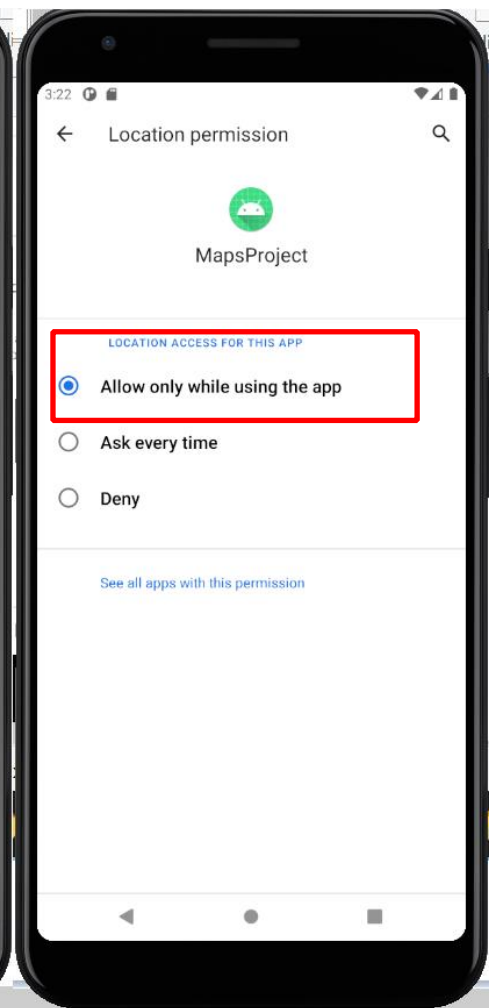
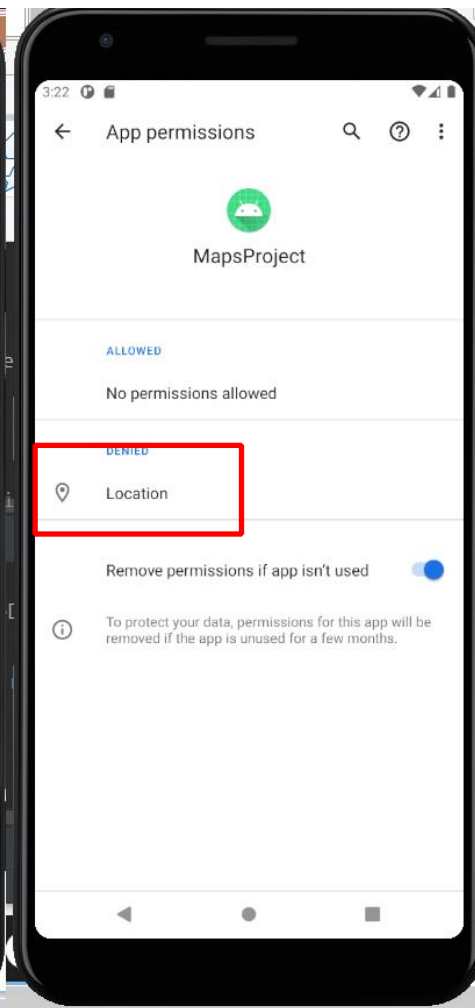
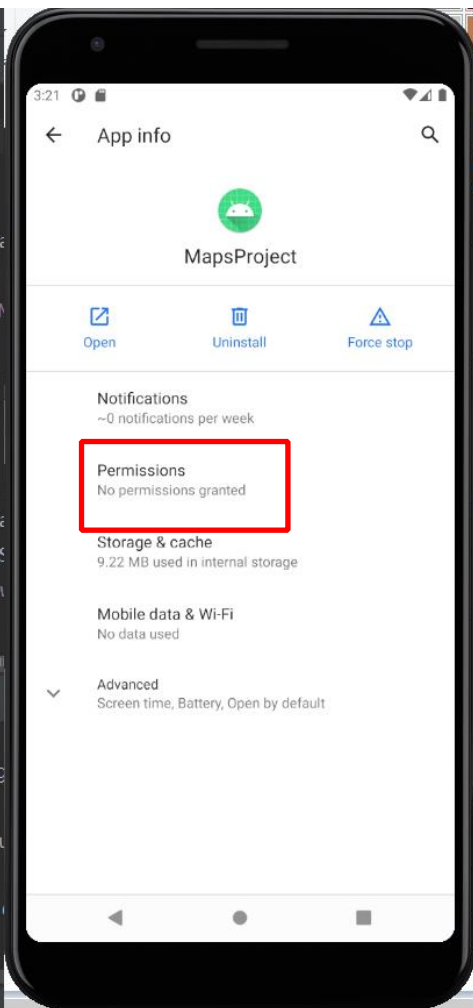
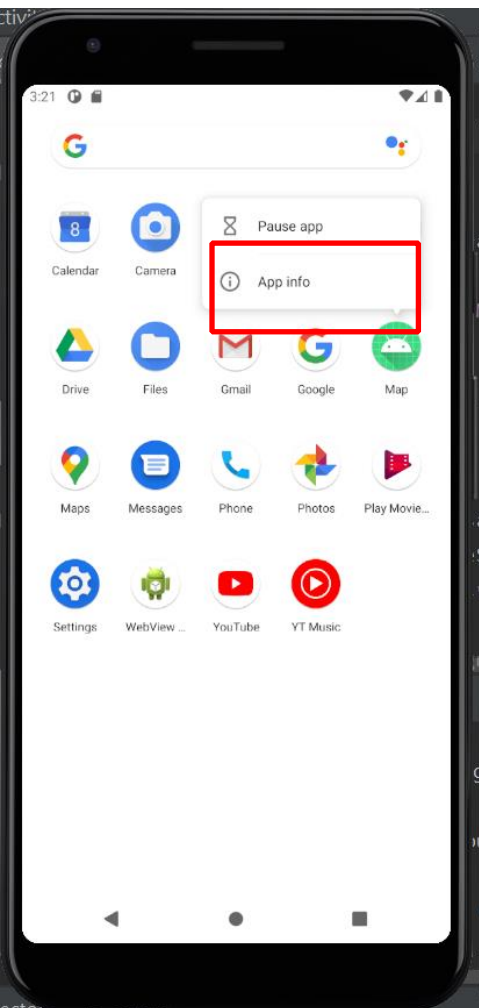
```
protected void onPause() {
    super.onPause();
    try {
        locationManager.removeUpdates(this);
    } catch (SecurityException e) {
        Log.e("Err", "No Location update permission remover");
    }
}

protected void onResume() {
    super.onResume();
    if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) == PackageManager.PERMISSION_GRANTED)
        locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 10,
Criteria.ACCURACY_COARSE, this);
}
```

# Run the app

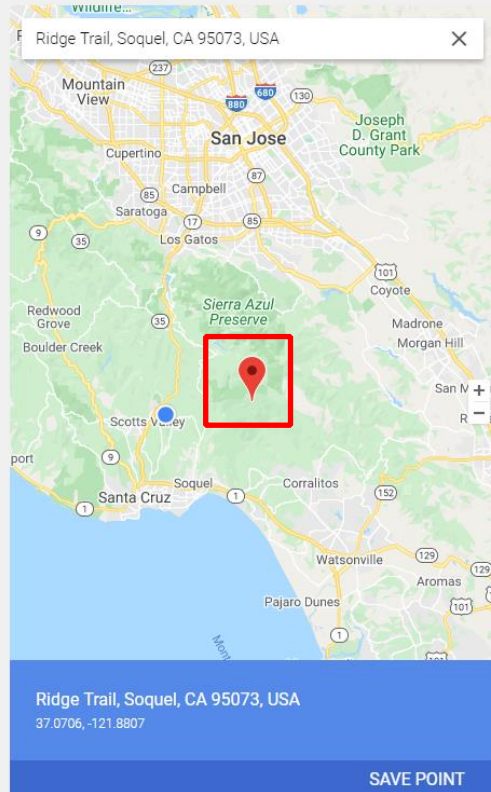






- Location
- Displays
- Cellular
- Battery
- Camera
- Phone
- Directional pad
- Microphone
- Fingerprint
- Virtual sensors
- Bug report
- Snapshots
- Record and Playback
- Settings
- Help

Single points Routes



IMPORT GPX/KML

Saved points

Points that you save shall appear here

SET LOCATION

