CMSC436: Programming Handheld Systems

Location & Maps

Today’s Topics

Location

Location support classes

Maps

Map support classes

Location Services

Mobile applications can benefit from being location-aware

Allows applications to determine their location and modify their behavior

Using Location Information

- Find businesses near the user’s current location
- Direct a user from a current location to a particular business
- Define a geofence
- Initiate action when user enters or exits the geofence

Location Architecture

Location

Permissions

FusedLocationProviderClient

LocationCallback

Location

Represents a position on the Earth

A Location instance consists of

- Latitude, longitude, timestamp
- Optionally: accuracy, altitude, speed, and bearing

Location Access Types

Category: Either foreground location or background location

Accuracy: Precise location or approximate location

Category

Foreground: app shares or receives location information only once, or for a defined amount of time

Background: app constantly shares location with other users or uses the Geofencing API

Accuracy

Approximate: Estimate typically accurate to 3km

Precise: Estimate typically accurate to 3m-50m

Permissions

- Background locations require
  - ACCESS_BACKGROUND_LOCATION
- Approximate accuracy requires
  - ACCESS_COARSE_LOCATION permission
- Precise accuracy requires
  - ACCESS_FINE_LOCATION permission
  - Should also request ACCESS_COARSE_LOCATION, because user can restrict accuracy

Types of Location Providers

Network – WiFi and cell tower

GPS - Satellite

Passive – Piggyback on the readings requested by other applications

LocationProvider Tradeoffs

GPS – expensive, accurate, slower, available outdoors

Network - cheaper, less accurate, faster, availability varies

Cached information – cheapest, fastest, not always available

FusedLocationProviderClient

Location-providing class that fuses different location providers

Part of Google Play Services

See: https://developers.google.com/android/guides/setup

FusedLocationProviderClient methods

getLastLocation()

getCurrentLocation()

requestLocationUpdates()

Requesting Location Updates

Create FusedLocationProviderClient

Create and configure a LocationRequest

Check device settings

Implement LocationCallback interface

Register for location updates

LocationCallback

Defines callback methods that are called when FusedLocationProviderClient location information changes

LocationCallback Methods

onLocationAvailability(LocationAvailability locationAvailability) : Unit

onLocationResult(locationResult: LocationResult): Unit

Recipe for Obtaining and Using Location Information

- Start listening for updates
- Maintain a "current best estimate" of location
- When estimate is “good enough”, stop listening for location updates
- Use best location estimate

Determining Best Location

Several factors to consider

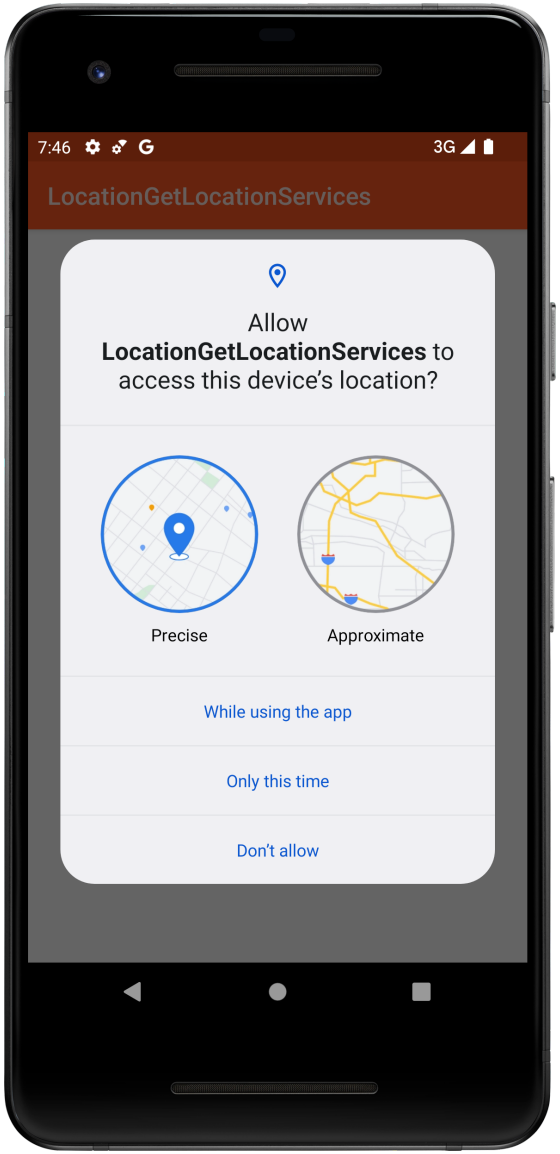
- Measurement time
- Accuracy
- Power usage

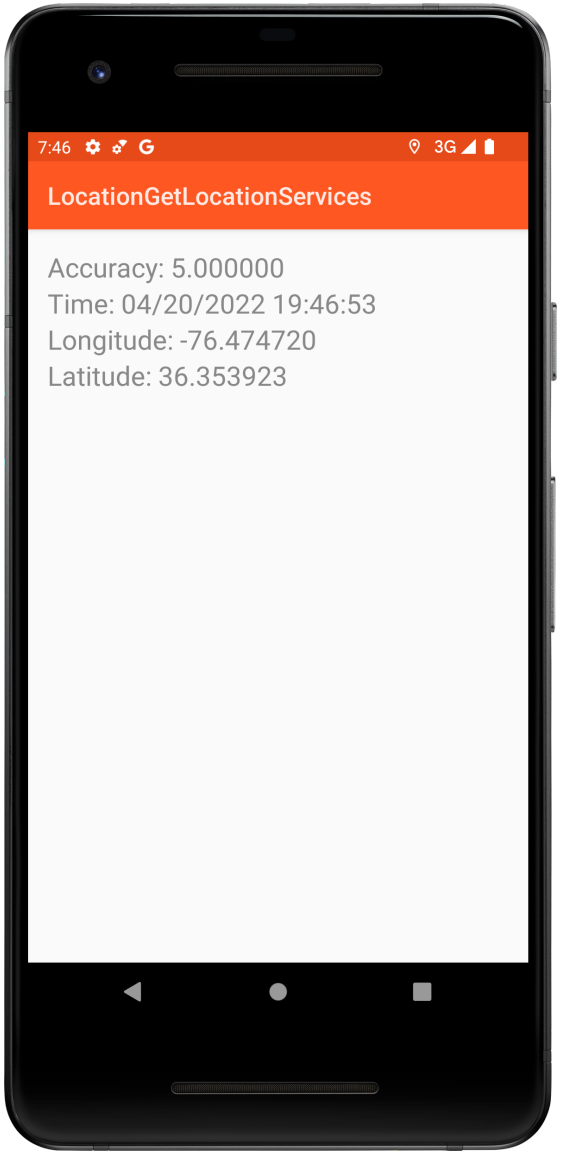
LocationGetLocationServices

Application acquires and displays the last known location

If necessary, acquires and displays new readings

LocationGet
LocationServices





Battery Saving Tips

Always check last known measurement

Return updates as infrequently as possible

Limit measurement time

Use the least accurate measurement necessary

Turn off updates in onPause()

Maps

A visual representation of area

Android provides Mapping support through the Google Maps Android API

Map Types

Normal – Traditional road map

Satellite – Aerial photograph

Hybrid – Satellite + road map

Terrain – Topographic details

Customizing the Map

Change the camera position

Add Markers & ground overlays

Respond to gestures

Indicate the user’s current Location

Some Map Classes

GoogleMap

MapFragment

Camera

Marker

Setting up a Maps Application

Set up the Google Play services SDK

Obtain an API key

Specify settings in Application Manifest

Add map to project

See: https://developers.google.com/maps/documentation/android/start

Map Permissions

```
<uses-permission android:name=
                        "android.permission.INTERNET"/>
<uses-permission android:name=
        "android.permission.ACCESS_NETWORK_STATE"/>
```

Map Permissions

```
<uses-permission android:name=
  "android.permission.WRITE_EXTERNAL_STORAGE"/>*

<uses-permission android:name=
        "com.google.android.providers.
                gsf.permission.READ_GSERVICES"/>
```

* For versions earlier than 8.3

Map Permissions

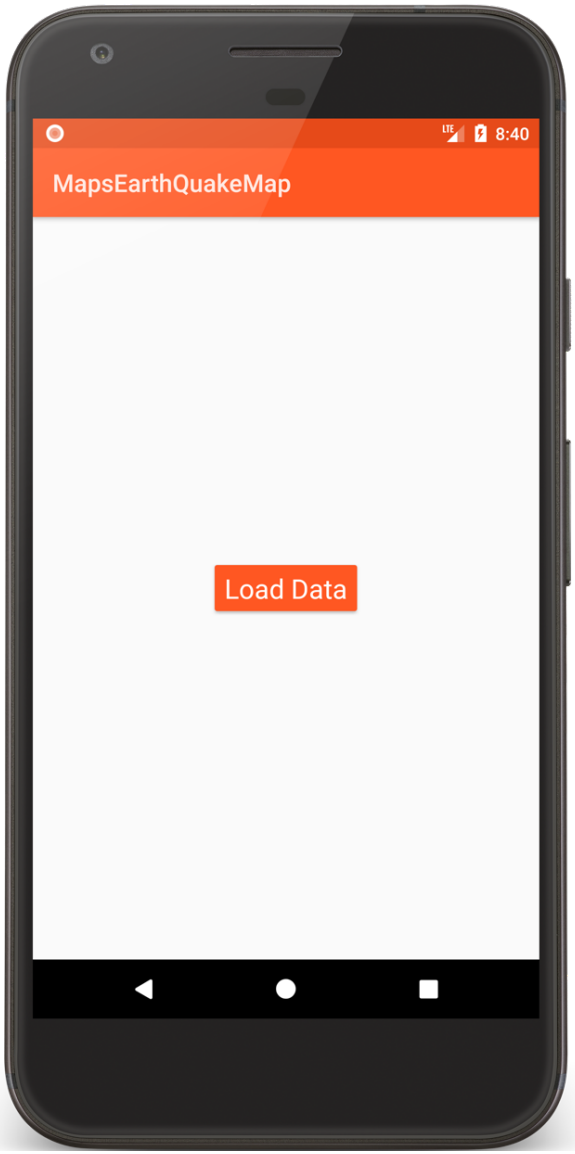
```
<uses-permission android:name=
   "android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name=
   "android.permission.ACCESS_FINE_LOCATION"/>
```

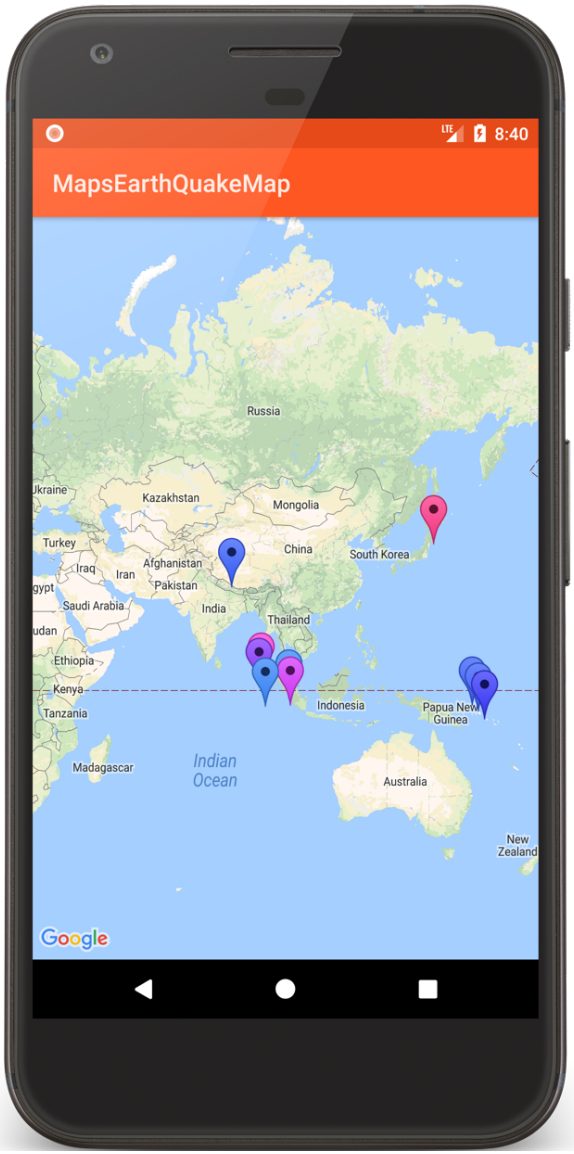
MapEarthQuakeMap

This application acquires earthquake data from a server

Then it displays the data on a map, using clickable markers

MapEarth
QuakeMap





MapsEarthquakeMapActivity.kt

```
// Set up UI and get earthquake data
public override fun onCreate(savedInstanceState: Bundle?) {
…
// The GoogleMap instance underlying the GoogleMapFragment defined
// in main.xml
 val map = supportFragmentManager.findFragmentById(R.id.map)
                                                    as SupportMapFragment?
    map?.getMapAsync(this)
}
```

MapsEarthquakeMapActivity.kt

```
// Called when Map is ready
override fun onMapReady(googleMap: GoogleMap) {
    mMapReady = true
    mMap = googleMap
    mMap!!.moveCamera(CameraUpdateFactory.newLatLng(
                                        LatLng(CAMERA_LAT,CAMERA_LNG)))
    if (mDataReady) {
        placeMarkers()
        mMapReady = false
    }
}
```

MapsEarthquakeMapActivity.kt

```
// Called when data is downloaded
override fun onDownloadfinished() {
    mDataReady = true
    if (mMapReady) {
        placeMarkers()
        mDataReady = false
    }
}
```

MapsEarthquakeMapActivity.kt

```
private fun placeMarkers() { // Add a marker for every earthquake
    for (rec in mRetainedFragment?.data!!) {
        // Add a new marker for this earthquake
        mMap!!.addMarker(MarkerOptions()
        // Set the Marker's position
                .position(LatLng(rec.lat,rec.lng))
        // Set the title of the Marker's information window
                .title(rec.magnitude.toString())
                // Set the color for the Marker
                .icon(BitmapDescriptorFactory.defaultMarker(
                        getMarkerColor(rec.magnitude))))
    }
}
```

Next Time

The ContentProvider Class

Example Applications

LocationGetLocationServices

MapEarthQuakeMap