

# CMSC436: Programming Handheld Systems

Fall 2017

# 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 stores near the user's current location

Direct a user from a current to a particular store

Define a geofence

Initiate action when user enters or exits the geofence

# Location Architecture

Location

LocationProvider

LocationManager

LocationListener

# Location

Represents a position on the Earth

A Location instance consists of:

Latitude, longitude, timestamp and, optionally, accuracy, altitude, speed, and bearing

# LocationProvider

Represents a location data source

Actual data may come from

- GPS satellites

- Cell phone towers

- WiFi access points



# LocationProvider Types

Network – WiFi and cell tower

GPS - Satellite

Passive – Piggyback on the readings requested by other applications

# NetworkProvider

Determines location based on cell tower and WiFi access points

Requires either

`android.permission.ACCESS_COARSE_LOCATION`

`android.permission.ACCESS_FINE_LOCATION`

# GPSProvider

Determines location using satellites

Requires

`android.permission.ACCESS_FINE_LOCATION`

# PassiveProvider

Returns locations generated by other providers

Requires

`android.permission.ACCESS_FINE_LOCATION`

# LocationProvider

Different LocationProviders offer different tradeoffs between cost, accuracy, availability & timeliness

# LocationProvider Tradeoffs

GPS – expensive, accurate, slower, available outdoors

Network - cheaper, less accurate, faster, availability varies

Passive – cheapest, fastest, not always available

# LocationManager

System service for accessing location data

```
getSystemService(Context.LOCATION_SERVICE)
```

# LocationManager

Determine the last known user location

Register for location updates

Register to receive Intents when the device nears or moves away from a given geographic area



# LocationListener

Defines callback methods that are called when Location or LocationProvider status changes

# LocationListener

void onLocationChanged (Location location)

void onProviderDisabled (String provider)

void onProviderEnabled (String provider)

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

# Obtaining and Using Location Information

Start listening for updates from LocationProviders

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

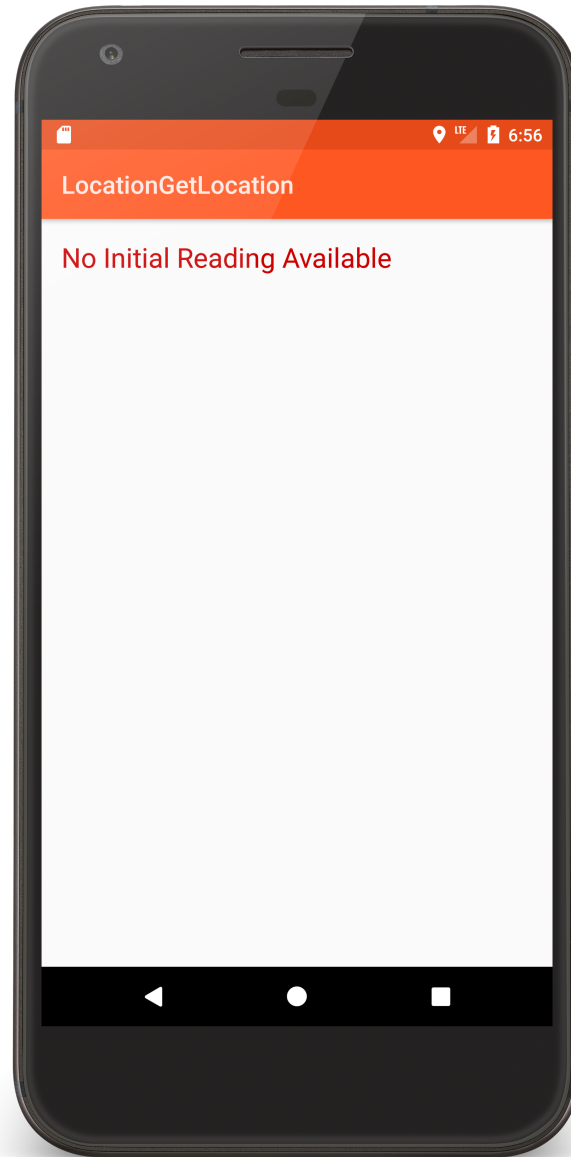
- Power usage

# LocationGetLocation

Application acquires and displays the last known locations from all providers

If necessary, acquires and displays new readings from all providers

Location  
Get Location



Extended controls

- Location
- Cellular
- Battery
- Phone
- Directional pad
- Microphone
- Fingerprint
- Virtual sensors
- Bug report
- Settings
- Help

GPS data point

Coordinate system: **Decimal**

Longitude: **44**

Latitude: **38.9847**

Altitude (meters): **0.0**

Currently reported location

Longitude: 44.0000  
Latitude: 38.9847  
Altitude: 0.0

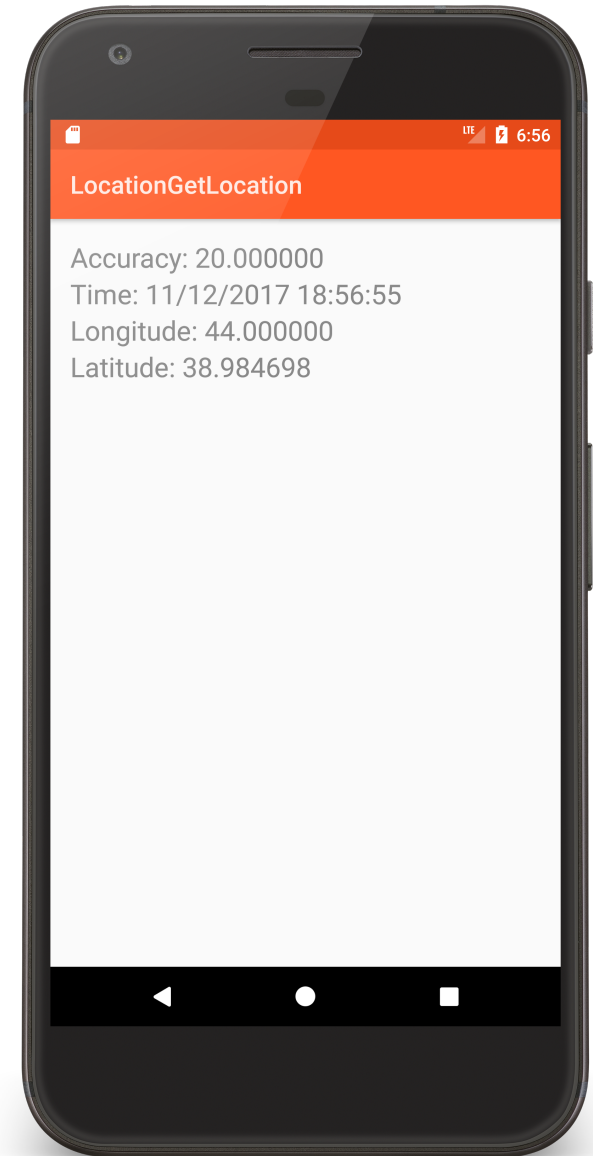
SEND

GPS data playback

Delay (sec)	Latitude	Longitude	Elevation	Name	Description

Speed 1X

LOAD GPX/KML



# LocationGetLocationServices

The same as LocationGetLocation, but uses newer FusedLocationProvider class

Uses Google Play Services



# 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 v2 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](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"/>
```



# Map Permissions

```
<uses-permission android:name=
```

```
"android.permission.ACCESS_COARSE_LOCATION  
"/>
```

```
<uses-permission android:name=
```

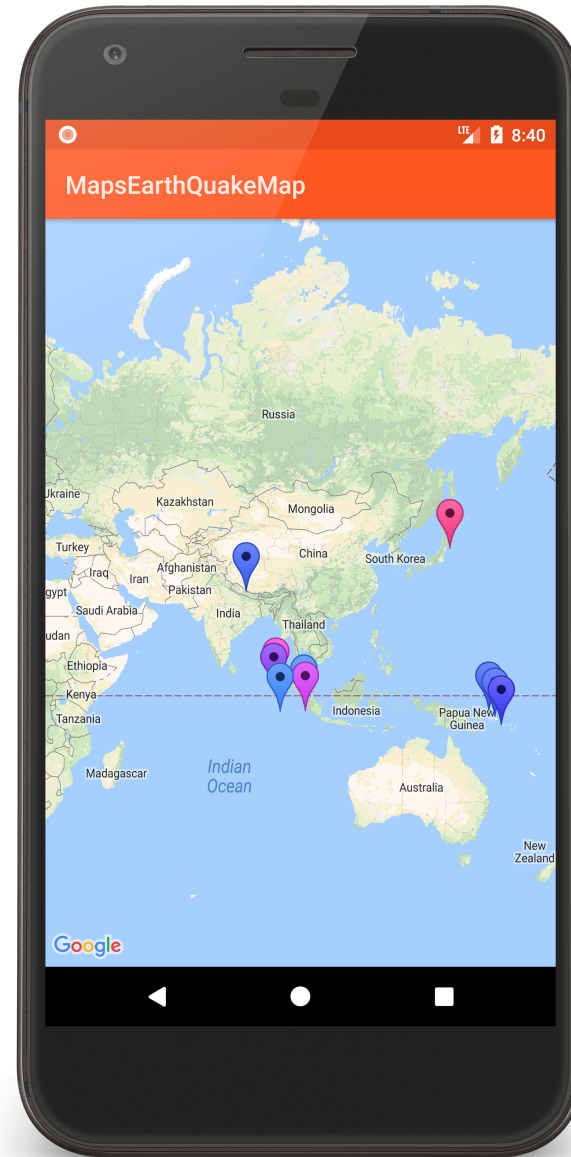
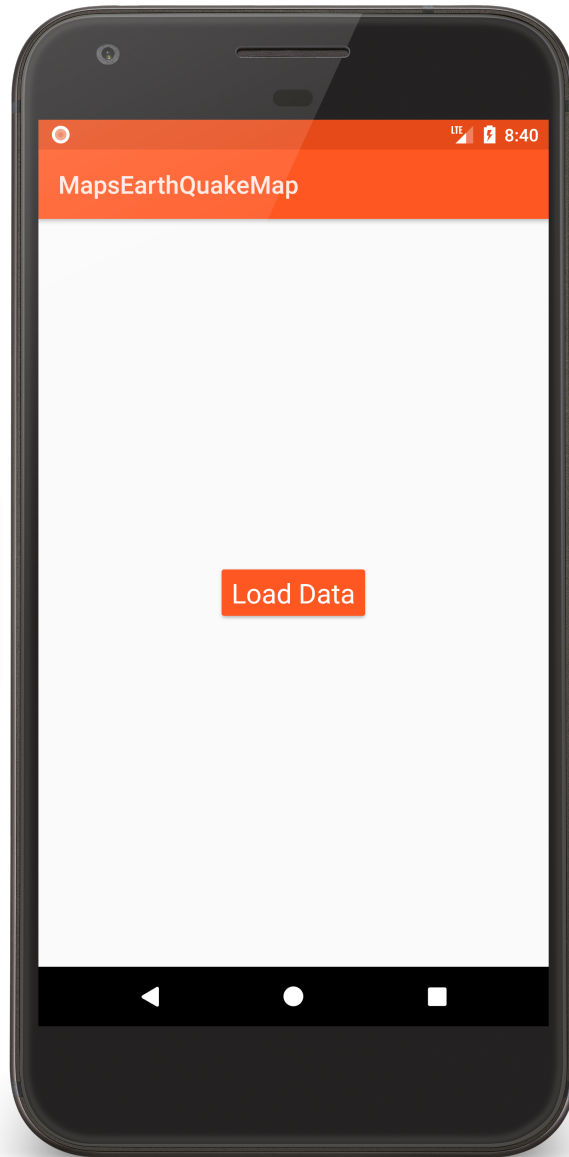
```
"android.permission.ACCESS_FINE_LOCATION"/>
```

# MapEarthQuakeMap

This application acquires earthquake data from a server

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

# MapEarth QuakeMap



```
public class MapsEarthquakeMapActivity extends Activity implements  
    RetainedFragment.OnFragmentInteractionListener, OnMapReadyCallback {  
    ...  
public void onCreate(Bundle savedInstanceState) {  
    // Set up UI and get earthquake data  
    ...  
    // The GoogleMap instance underlying the GoogleMapFragment defined in main.xml  
    MapFragment map = ((MapFragment) getSupportFragmentManager()  
        .findFragmentById(R.id.map));  
    map.getMapAsync(this);  
}
```

*// Called when Map is ready*

```
public void onMapReady(GoogleMap googleMap) {
```

```
    mMapReady = true;
```

```
    mMap = googleMap;
```

```
    mMap.moveCamera(  
        CameraUpdateFactory.newLatLng(new LatLng(CAMERA_LAT, CAMERA_LNG)));
```

```
    CameraUpdateFactory.newLatLng(new LatLng(CAMERA_LAT, CAMERA_LNG)));
```

```
    if (mDataReady) {
```

```
        placeMarkers();
```

```
        mMapReady = false;
```

```
    }
```

```
}
```

```
// Called when data has been downloaded  
public void onDownloadfinished() {  
    mDataReady = true;  
    if (mMapReady) {  
        placeMarkers();  
        mDataReady = false;  
    }  
}
```

```
private void placeMarkers() {  
    // Add a marker for every earthquake  
    for (EarthquakeRec rec : mRetainedFragment.getData()) {  
        // Add a new marker for this earthquake  
        mMap.addMarker(new MarkerOptions()  
            // Set the Marker's position  
            .position(new LatLng(rec.getLat(), rec.getLng()))  
            // Set the title of the Marker's information window  
            .title(String.valueOf(rec.getMagnitude()))  
            // Set the color for the Marker  
            .icon(BitmapDescriptorFactory.defaultMarker(  
                getMarkerColor(rec.getMagnitude()))));  
    }  
}
```

# Next Time

## The ContentProvider Class