Lecture
Location Based services

DT228/3

Dr. Susan McKeever

Location Awareness

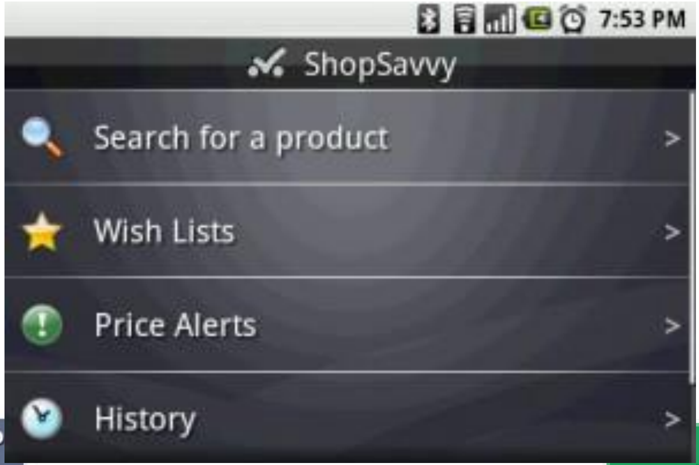
Mobile Devices – location changes constantly

Phone has sensors to detect its own location

Enables many functionality

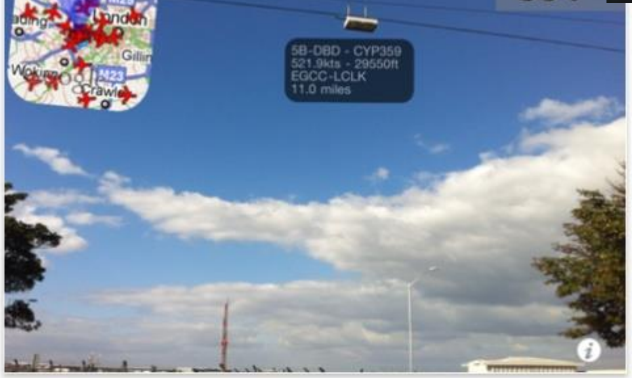
- What am I near?
- Who’s nearby?
- How far away is..?
- What route did I run?

Examples of Android Apps that make use of location awareness



Plane finder

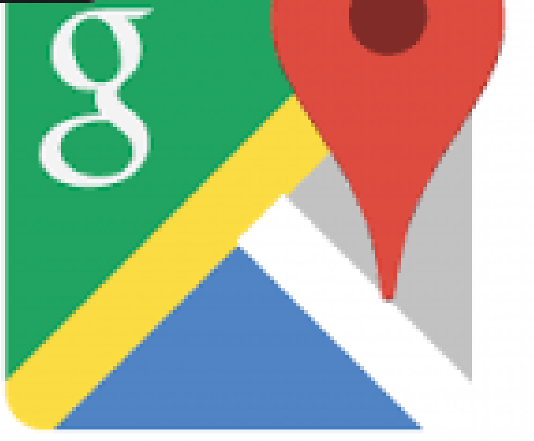
Identify any plane in the sky



Shop Savvy

ShopSavvy lets you compare prices in local and online stores.. Just scan and check

Google maps



To put Location into your own app

- Need to know WHERE your phone is (phone needs to detect its own location)
  - Put location detection into your code…

- Need to use that location to do something useful..

Location Based services

- Various ways a phone can detect it’s own location
  - GPS – latitude, longitude
  - Known Wifi hotspots
  - Cell tower triangulation

- Android provides `LocationProvider` objects that you can hook into
  - E.g. GPS is a Location provider

- To choose a provider - consider: cost? Accuracy? Frequency of update? Battery impact?

Android: Steps to getting your app to detect device location

1. Get hold of the **Location Manager**
   - System Utility class that can report the device's current location
   - Retrieved using the `getSystemService()` method of your activity (Note; this method can be used getting at various system services.. E.g. Power manager, UI modes, Search manager, Vibration mode etc.
2. Request location updates from a **location provider**
   - E.g. `requestLocationUpdates()` method of your activity
   - Other methods available too... ...e.g. `getLastKnownLocation(); getBestProvider()`

Android: Steps to getting your app to detect device location

3. Implement the **LocationListener**

   - Listen for location updates if the app is supposed to “do” something every time location change detected

4. Implement the callback methods from the listener

   - Various methods
   - Most common is

```
        onLocationChanged(Location location)
        ...
```

Code Sample

- LocationTest1

- Basic example of how to display the current location.

  - Follows through the steps described..
    - (1), (2), (3), (4)

- Testing through the emulator, need to enter in the “coordinates” of the phone

- Or else test your phone outside..

Things to note about the code...

- The `requestLocationUpdates()` method has a bunch of important things to control about how you receive location updates –

- Different method signatures if you look at the API:

https://developer.android.com/reference/android/location/LocationManager

- Look at the parameters from your code
  - `provider` Which Location provider to use? GPS_SERVICE
  - `minTime:` How often to update on location? 60 milliseconds
  - `minDistance`: How much distance you have to have travelled before updates notified ..5 metres

    How should these be used.. E.g. For a navigation app used in the car versus walking? For a

Things to note about the code...

A careful choice of these values can make a difference to power consumption and to how your application works:

1.minTime – Wise choice of value for *minTime* will help you reduce power consumption.

Elapsed time between location updates will never be less than *minTime*, but it can be greater, because it is influenced by the implementation of each given **Provider** and the update interval requested by other running applications.

2.minDistance – Unlike *minTime* this parameter can be turned off by setting its value to 0. However, if *minDistance* is set to a value greater than 0, location provider will only send updates to your application *if location has changed at least by given distance.*

This parameter is not a great power saver like *minTime*, but it should be kept in mind although.

Beware of the battery…

- Battery life limited
- Location updates **very**
  Power hungry
- Each provider – GPS, Wifi, Cell based location – separately absorbing battery life
- Only check location when your activity **needs** to. e.g. Hidden activity doesn’t.
- Use lifecycle methods to stop and start location updates - see code sample:
  - `onPause()` - stop updates
  - `onResume()` – start up updates again

Permissions

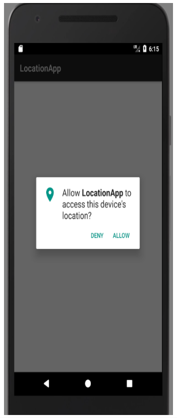
App needs permissions to protect user privacy e.g.
sending SMS, location tracking, contacts access

Normal permissions

Low risk - e.g. Permission to send SMS- Just set in Android Manifest file – checked at app install time

Dangerous

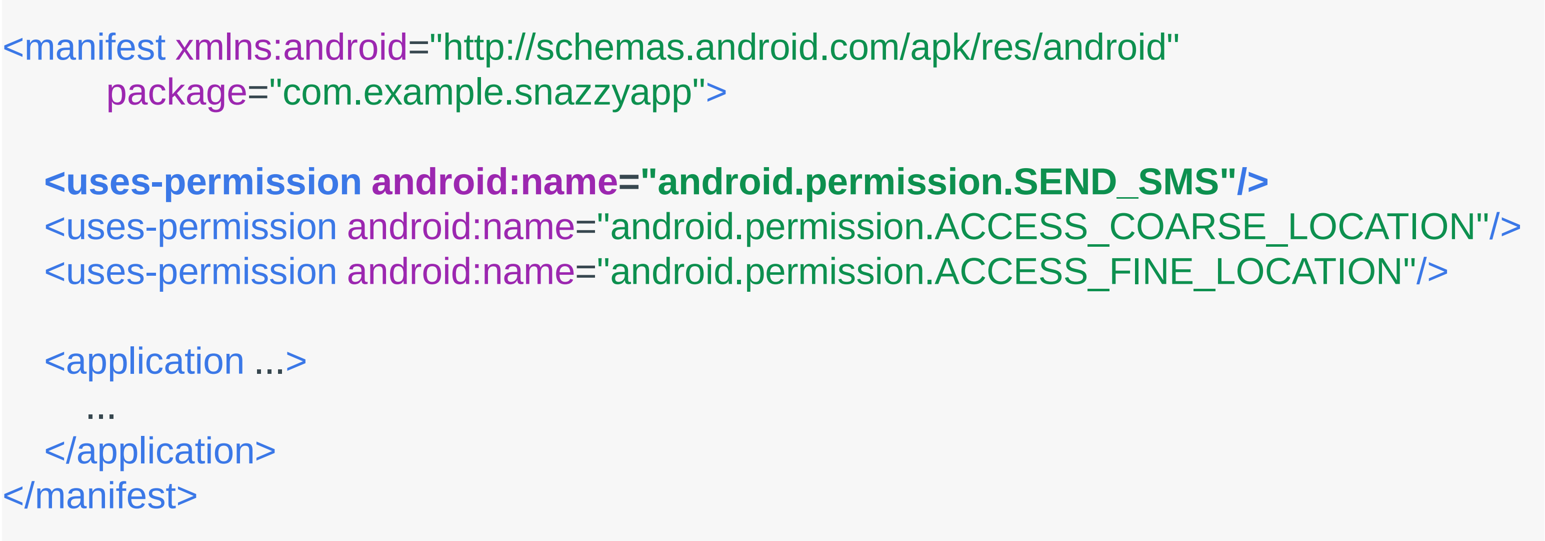
Higher risk permissions - Set in Android Manifest file AND also check at run time
(note: this run time checking model came in in API 23)



Allow **LocationApp** to access this device's location?

Permissions

Manifest file:
<uses-permission> tag

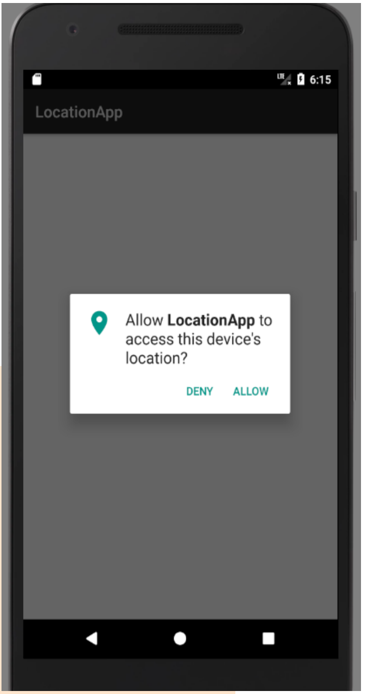


```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="com.example.snazzyapp">

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

    <application ...>
        ...
    </application>
</manifest>
```

Both normal and dangerous permissions must be declared here

Permissions

Only dangerous permission need this step
e.g. for location tracking:



Allow **LocationApp** to access this device's location?

```
// if your app doesn't have permission for location tracking

if (ContextCompat.checkSelfPermission(MainActivity.this,
        Manifest.permission.ACCESS_FINE_LOCATION)
        != PackageManager.PERMISSION_GRANTED)
{

// Display the dialog box to request it

    ActivityCompat.requestPermissions(MainActivity.this,
            new String[]{Manifest.permission.ACCESS_FINE_LOCATION},
            MY_PERMISSION_GPS);
}
   else
{ // did have permission so just go ahead and start location tracking
```

Only dangerous permission need this step

Location tracking permission

- (If you don’t, security violation error not well flagged.. Difficult to diagnose!)

- Two `AndroidManifest.XML` permissions involved:

1. **ACCESS_COARSE_LOCATION**
   - E.g. Cell_ID, Wifi – 1K-ish
   - `<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />`

2. **ACCESS_FINE_LOCATION**
   - More specific e.g. 10m
   - E.g. GPS

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

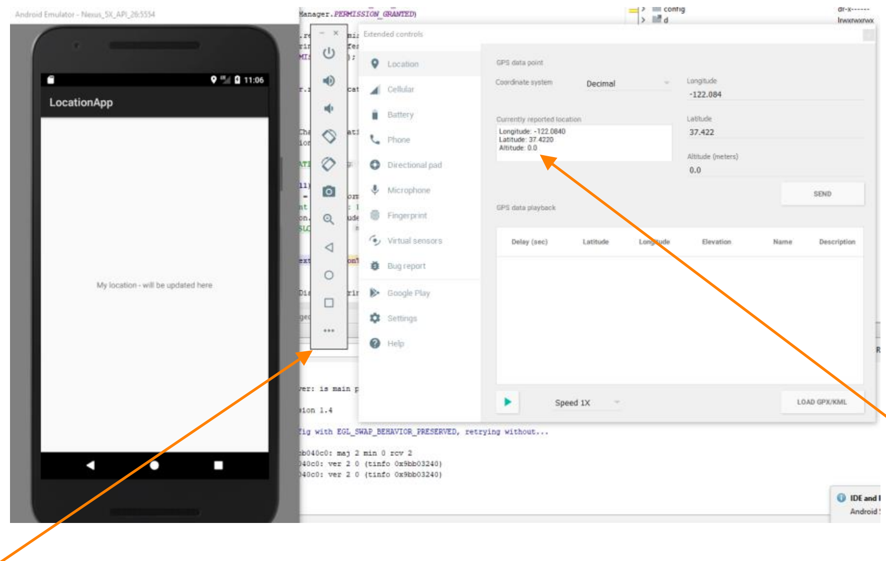
Another Example..

LocationTest2 example...

- What’s it doing?

Emulator: Mocking up location

- It’s not a real device
- Need to artificially feed in mocked up GPS coordinates
- This tests the location tracker, location listener etc.



My location - will be updated here

GPS data point

Coordinate system: Decimal

Longitude: -122.084

Latitude: 37.422

Altitude (meters): 0.0

Currently reported location

Longitude: -122.0840
Latitude: 37.4220
Altitude: 0.0

GPS data playback

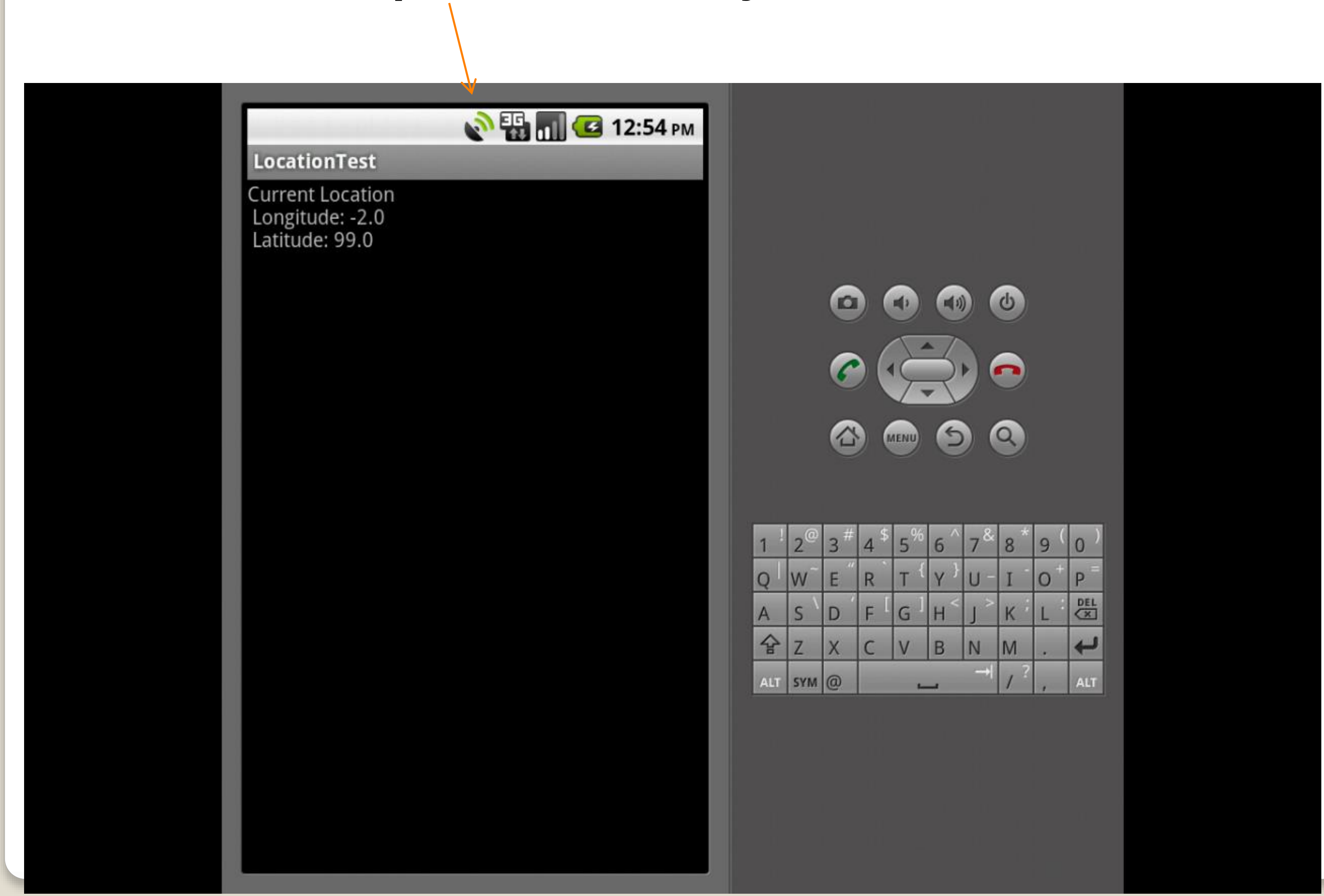
Delay (sec)	Latitude	Longitude	Elevation	Name	Description

```
Manager.PERMISSION_GRANTED)
ver: is main p
sion 1.4
fig with EGL_SWAP_BEHAVIOR_PRESERVED, retrying without...
bb040c0: maj 2 min 0 rcv 2
040c0: ver 2 0 (tinfo 0x9bb03240)
040c0: ver 2 0 (tinfo 0x9bb03240)
```

In your emulator, click the …

Then type in your own GPS coordinates

You’ll know you’re connected to a location provider when you see this icon

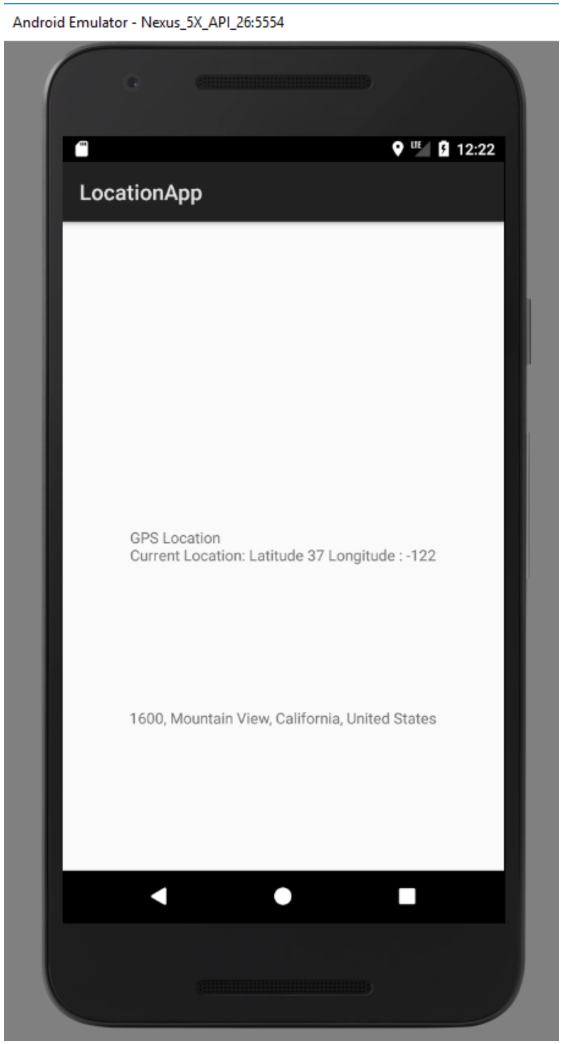


12:54 PM

LocationTest

Current Location
Longitude: -2.0
Latitude: 99.0

Geocoding



Android Emulator - Nexus_5X_API_26:5554

GPS Location
Current Location: Latitude 37 Longitude : -122

1600, Mountain View, California, United States

Longitude: -122.0840
Latitude: 37.4220

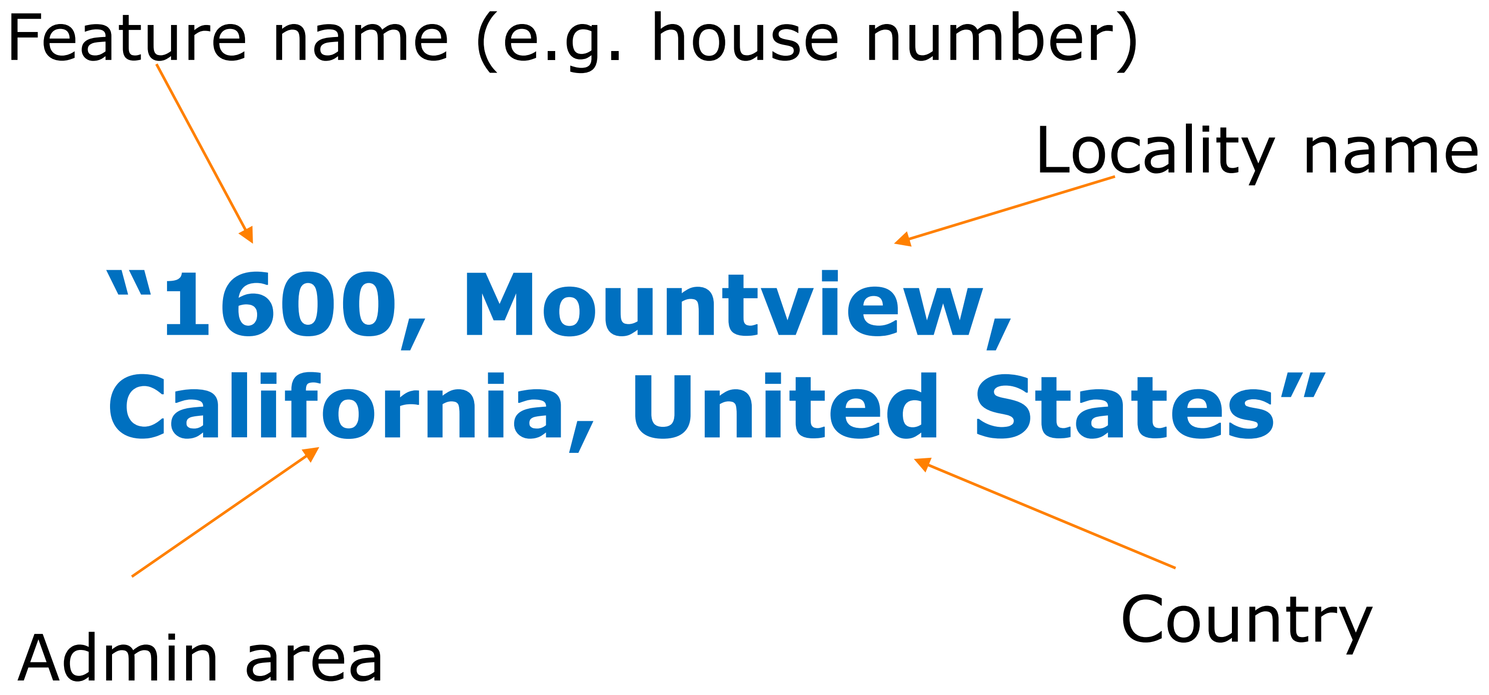
Reverse Geocoded to:

“1600, Mountview, California, United States”

Geocoded

Longitude: -122.0840
Latitude: 37.4220

Geocoding



Feature name (e.g. house number)

Locality name

“1600, Mountview, California, United States”

Admin area

Country

Geocode class

To use geocoding, get a GeoCode object of the Geocode class

Geocoder(Context context, Locale locale)
Constructs a Geocoder whose responses will be localized for the given Locale.

Then, pass in the latitude, longitute to the getFromLocation method of this class

getFromLocation(double latitude, double longitude, int maxResults)
Returns an array of Addresses that are known to describe the area immediately surrounding the given latitude and longitude.

Sample code

```
// get a geocoder object
Geocoder geo = new Geocoder(MainActivity.this.getApplicationContext(),
Locale.getDefault());

// call the method to translate GPS to address(es)
List<Address> addresses = geo.getFromLocation(location.getLatitude(),
location.getLongitude(), 1);

if (addresses.size() > 0)
{
            String addressName  = addresses.get(0).getFeatureName() + ", " +
                    addresses.get(0).getLocality() + ", " +
                    addresses.get(0).getAdminArea() + ", " +
                    addresses.get(0).getCountryName();


}
```

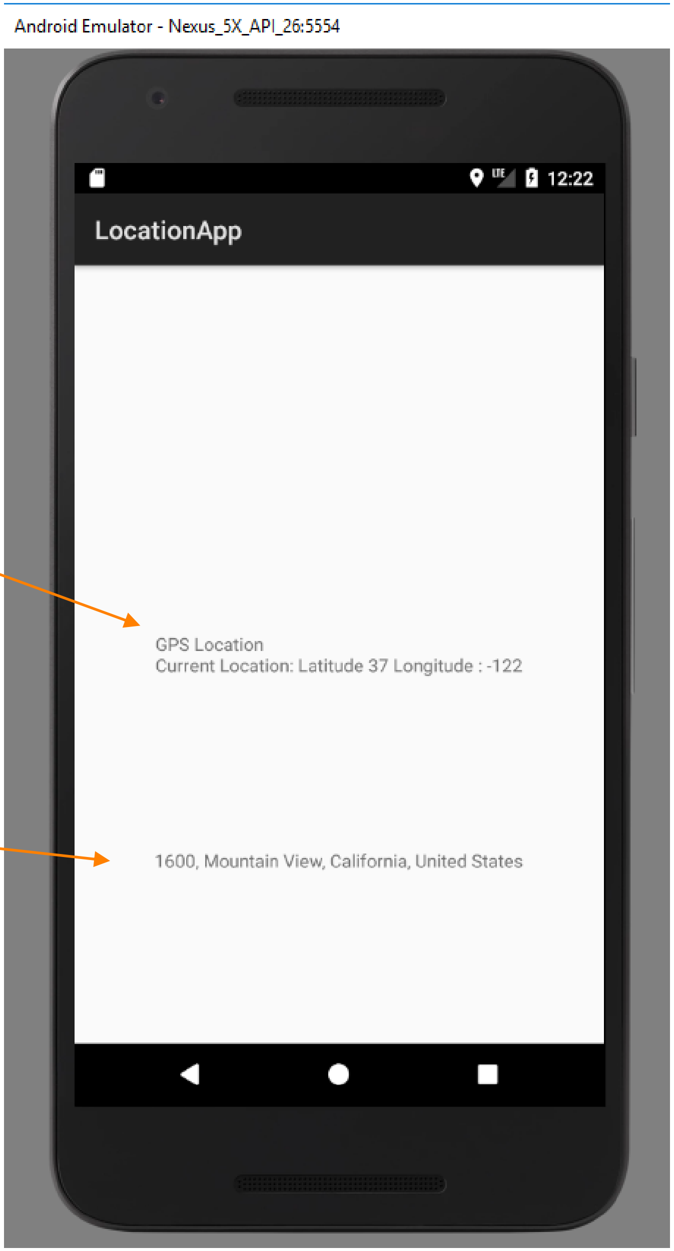
Note: try/ catch loop not included

Sample code

Longitude: -122.0840
Latitude: 37.4220

(Reverse) geocoded into

“1600, Mountview, California, United States”



Android Emulator - Nexus_5X_API_26:5554

LocationApp

GPS Location
Current Location: Latitude 37 Longitude : -122

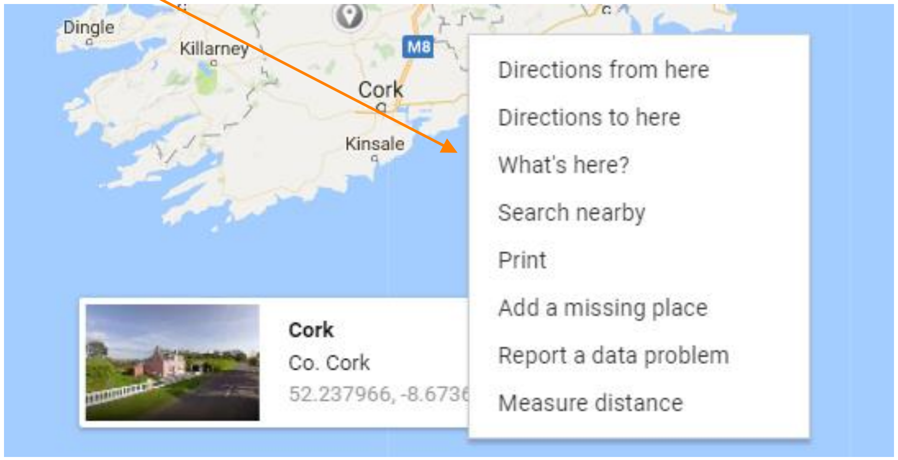
1600, Mountain View, California, United States

How to find GPS coordinates of a place

Go to the location on google maps

https://www.google.ie/maps?source=tldsi&hl=en

Right click and select “what’s here”



Summary

- Location aware app examples

- Implementing location awareness

  - Location providers – GPS, Wifi, Cell
  - LocationManager
  - Criteria/ Best provider..
  - minTime/ maxtime

- Battery life – use onPause() and onResume()

- Emulator - mocking up location
- GeoCoding