

CS371m - Mobile Computing

Location

(Location, Location, Location)

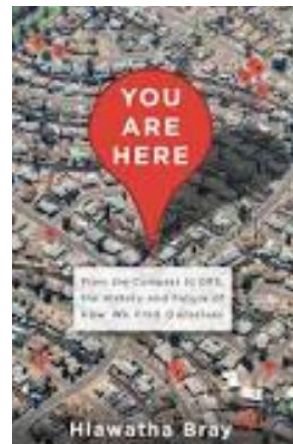
Cheap GPS



<http://xkcd.com/407/>

Android and Location

- inputs to location for Android device include:
- GPS
- cell-ID (cell tower)
- Wi-Fi networks
 - Network Location Provider combines cell-ID and Wi-Fi data
- Good reference for history of location:
You Are Here: From the Compass to GPS, the History and Future of How We Find Ourselves



Location is Important

"On 8 January 2005 at 02:43 GMT, the *USS San Francisco* collided with an [undersea mountain](#) about 675 kilometers (364 nautical miles, 420 statute miles) southeast of [Guam](#) while operating at flank (maximum) speed at a depth of 525 feet."

- Wikipedia article on the USS San Francisco, SSN - 711

- Dead reckoning
- radar fix
- visual fix
- Loran
- Omega
- Navsat
- GPS
- Active Sonar
- Inertial Navigation System

Location, Location, Location



USS San Francisco -

<http://tinyurl.com/l6vuucm>



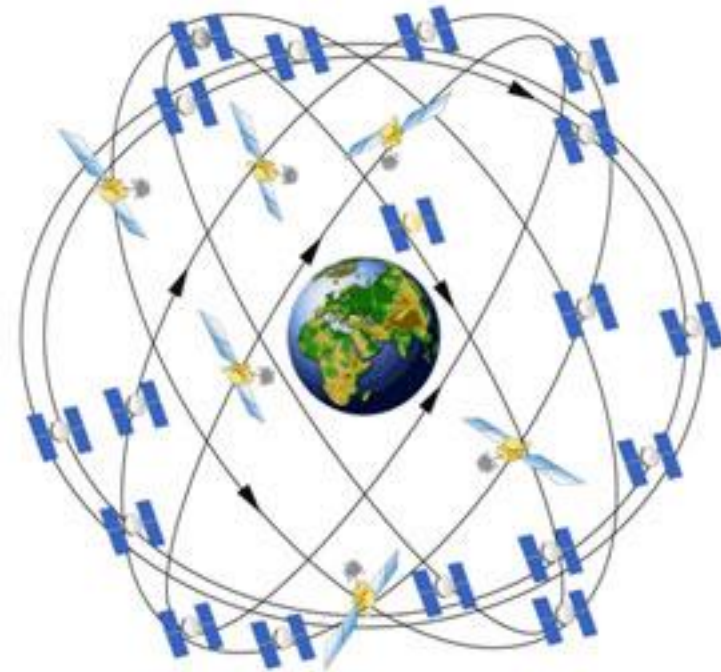
Global Positioning System

- GPS
- US System that provides position, navigation, and timing
- Space Segment, Control Segment, User Segment
- US Air Force develops, maintains, and operates the space segment and control segment



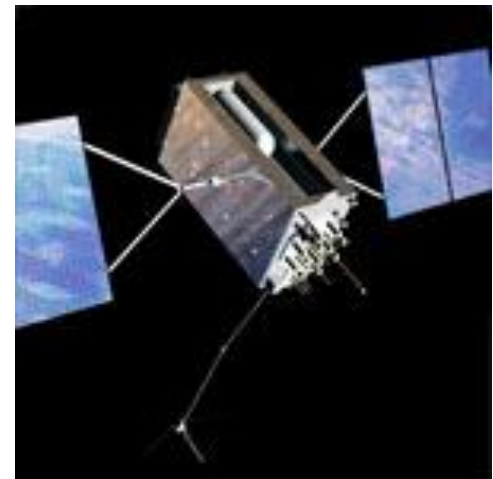
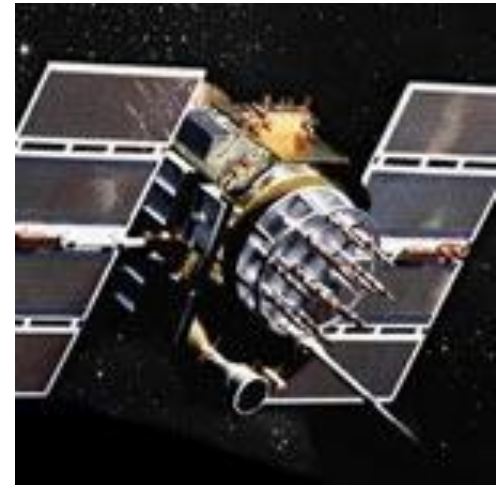
GPS Space Segment

- 24 core satellites
- medium earth orbit, 20k km above the earth
- 6 orbital planes with 4 satellites each
- generally 4 satellites in line of sight at any spot on the earth
- recently upgraded to 27 sats



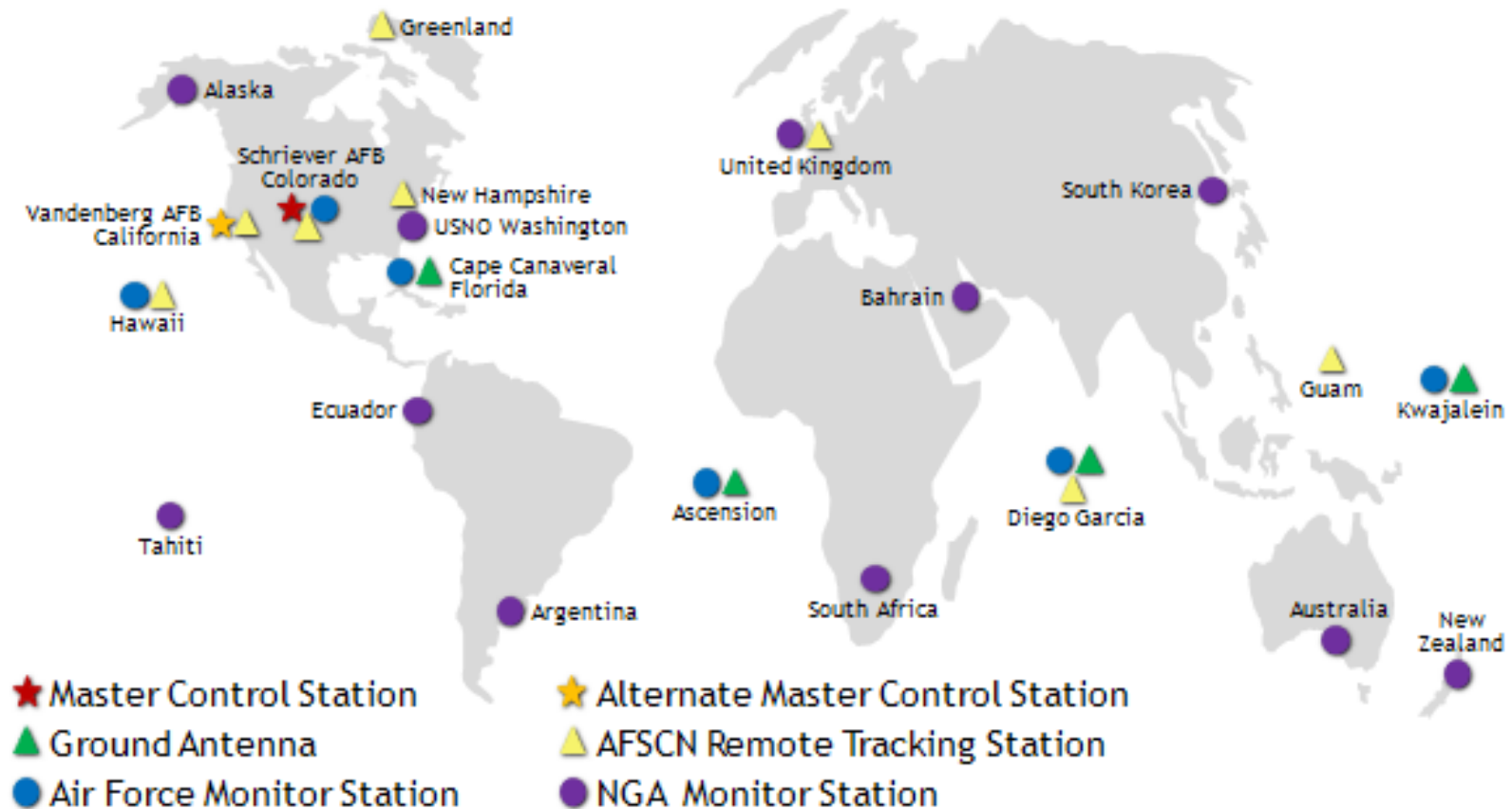
GPS Space Segment

- satellites circle the earth twice a day
- upgraded over time with different generations of satellites
- Current generation of satellites being developed by Lockheed - Martin (FOCS)



GPS Control Segment

- Ground facilities that
 - monitor transmissions, perform analysis, and send commands and data to satellites



GPS User Segment

- Onboard clocks with accuracy of 1 nanosecond (1 billionth of a second)
- Satellites transmit one-way
- receiver calculates position and course by comparing time signals from multiple satellites with the known position of those satellites

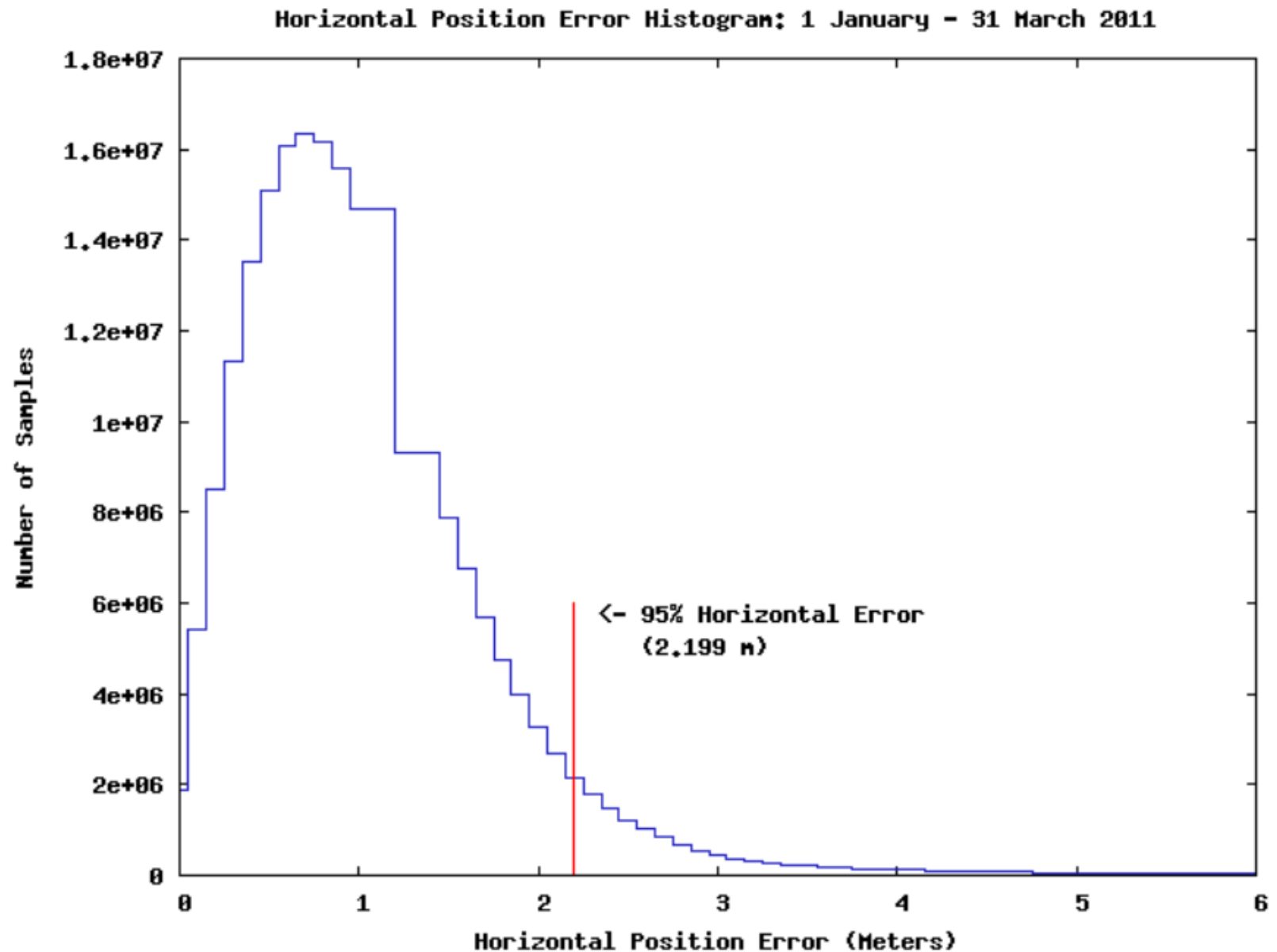
CRAZY PHENOMENON		IF IT WORKED, COMPANIES WOULD BE USING IT TO MAKE A KILLING IN...	ARE THEY?
REMOTE VIEWING	OIL PROSPECTING		
DOWSING			
AURAS	HEALTH CARE COST REDUCTION		
HOMEOPATHY			
REMOTE PRAYER			
ASTROLOGY	FINANCIAL/BUSINESS PLANNING		
TAROT			
CRYSTAL ENERGY	REGULAR ENERGY		
CURSES, HEXES	THE MILITARY		
RELATIVITY	GPS DEVICES		✓
QUANTUM ELECTRODYNAMICS	SEMICONDUCTOR CIRCUIT DESIGN		✓

EVENUALLY, ARGUING THAT THESE THINGS WORK MEANS ARGUING THAT MODERN CAPITALISM ISN'T *THAT* RUTHLESSLY PROFIT-FOCUSED.

GPS User Segment

- accuracy normally within 5 - 10 meters
- precision requires accuracy of clocks and timing signal on the order of 20 nanoseconds
- the Special and General theories of Relativity must be taken into account to achieve the desired accuracy
- Special relativity predicts clocks on satellites go slower, on the order of 10 microseconds per day
- General relativity predicts the mass of the earth will have an effect

GPS Accuracy



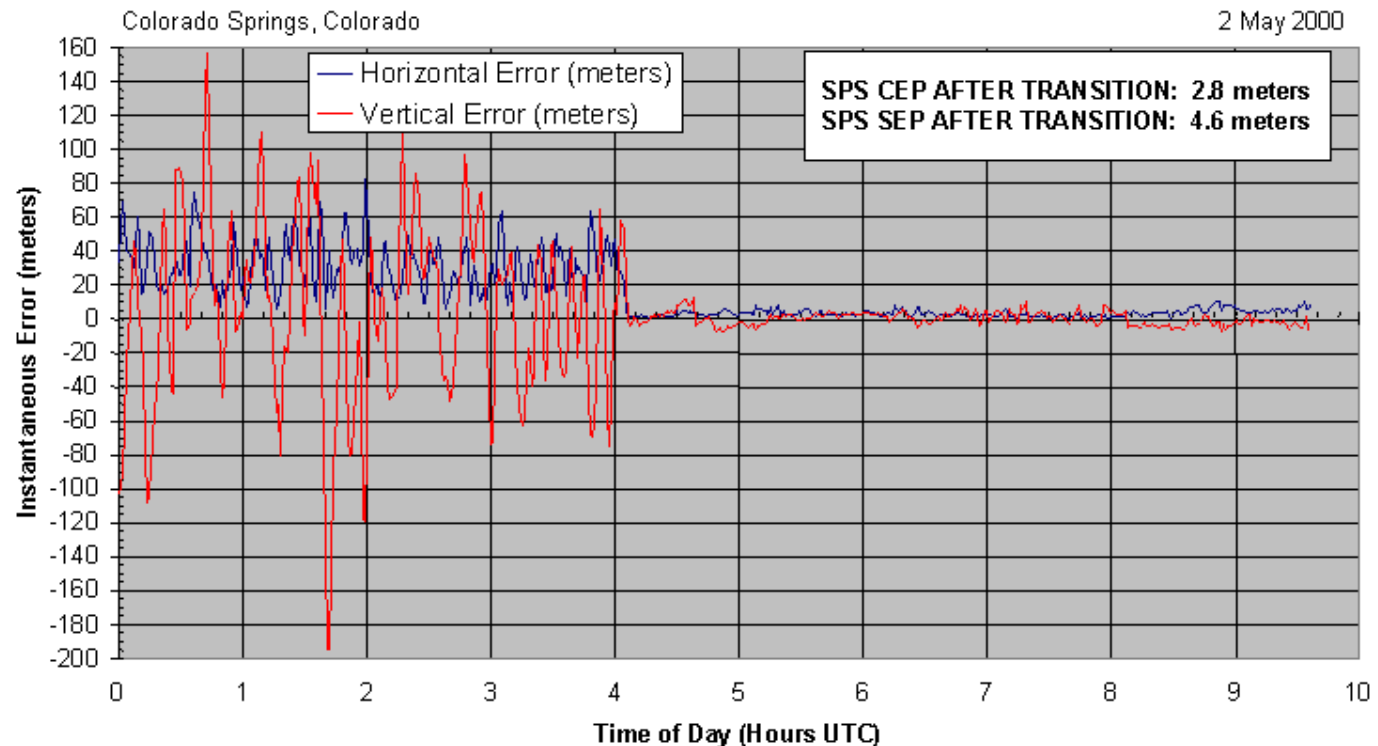
GPS Accuracy

- Selective Availability: intentional degradation of signals for civilian use

— ended
in 2000



SA Transition -- 2 May 2000



GPS Accuracy

- civilian GPS: aka SPS
- military GPS: aka PPS
- military broadcasts on two frequencies, civilian only one
- "This means military users can perform *ionospheric correction*, a technique that reduces radio degradation caused by the Earth's atmosphere. With less degradation, PPS provides better accuracy than the basic SPS. "



ANDROID AND LOCATION

Android and Location

- Currently 3 methods of obtaining location
- GPS
- NETWORK
 - combines cell tower triangulation and wireless networks
- PASSIVE
 - not a real provider, just piggy back off other applications
 - similar to software sensors

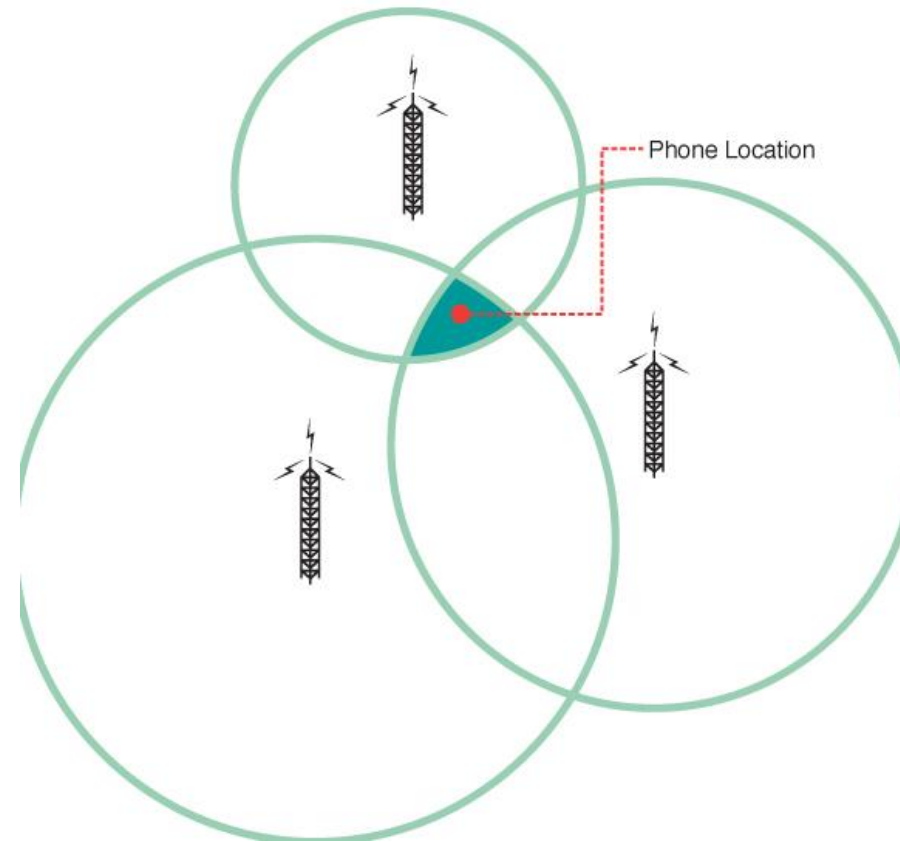
GPS

- most accurate but,
- only works OUTDOORS
- quickly consumes battery power
- delay in acquiring satellites or re- acquiring if lost



Network

- Combines cell tower triangulation and in range wireless networks
- If no cellular capability or plan on device (tablets?) then just wireless networks



Clicker Question

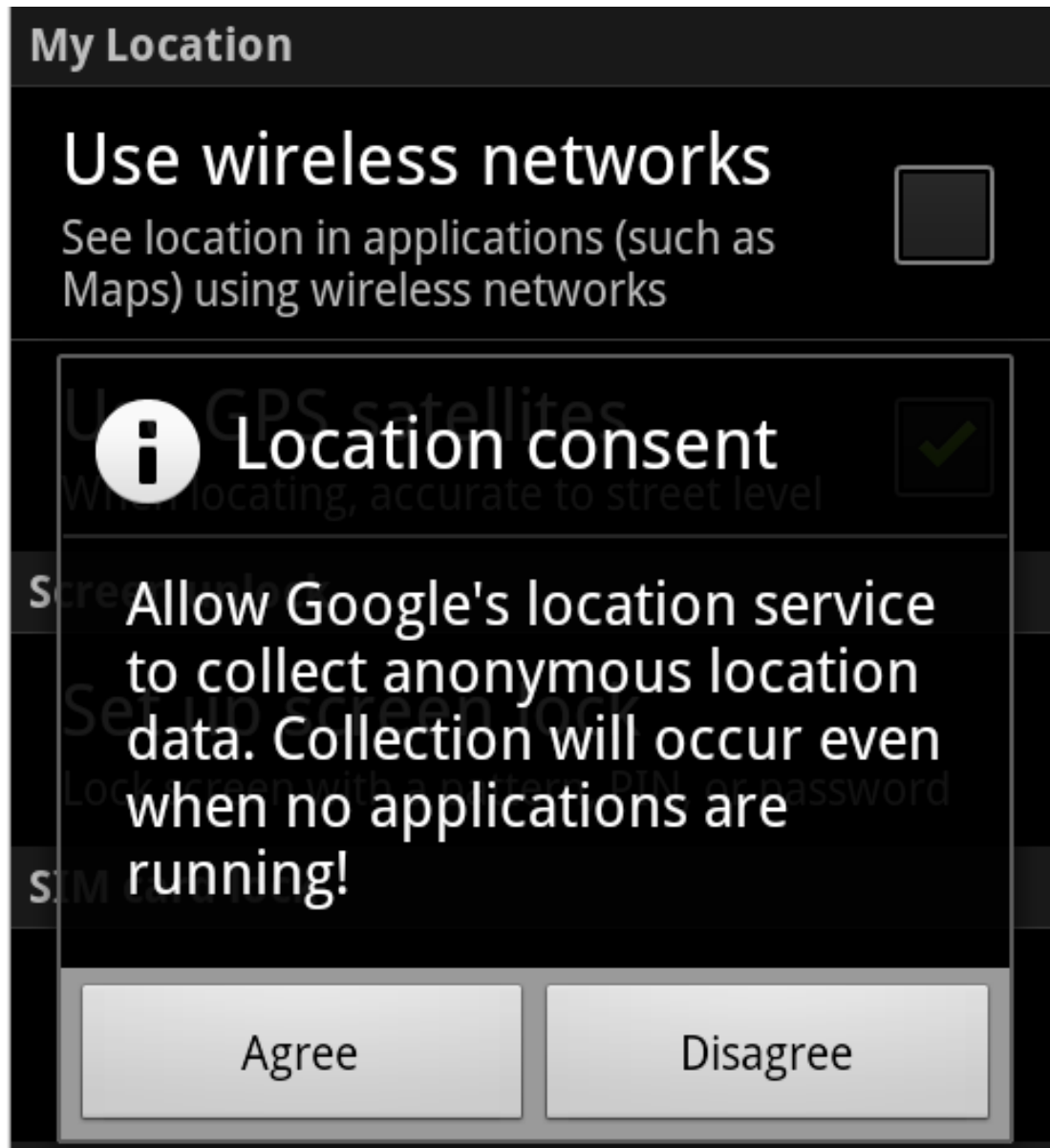
- Do you think of Google as a “creepy” company?
 - A. No
 - B. Yes
 - C. Undecided

Wireless Network and Location

- Formerly used StreetView cars
- Now, use the devices themselves to map locations to wifi spots
- Many other companies (Apple and Microsoft) do the same thing
- default on dev phones was checked



Google Location Services



Getting Location

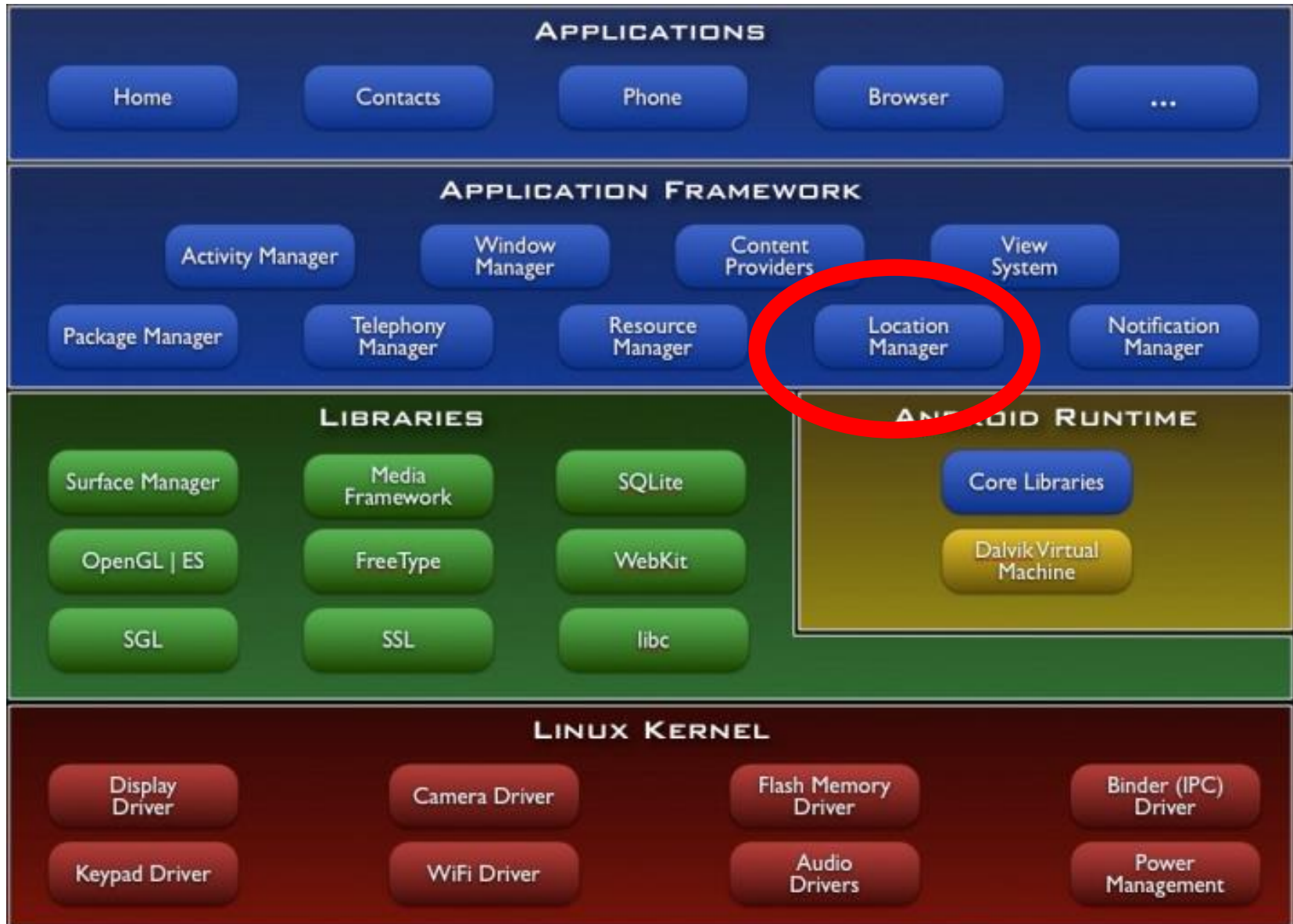
- Obtaining location:
- Demo older, detailed approach
 - Flexible, detailed, but developers typically make mistakes
 - Consume lots of battery power, GPS receiver requires large amounts of power to amplify signal from satellites.
- Newer, higher level, more abstract approach using Google Play Services

FINDING LOCATION VIA THE LOCATION-MANAGER

Finding Location

- Add appropriate permission to `AndroidManifest.xml`
- Get instance of `LocationManager` using `getSystemService` method using `LOCATION_SERVICE`
- Choose location provider (from all providers or using `getBestProvider` method)
- Implement a `LocationListener` class
- Call `requestLocationUpdates` method with chosen provider so `LocationListener` start receiving location information

LocationManager



Quickly Finding Location

- If you want a simple fix (location) get the LocationManager and ask for the last, best known position

```
mgr = (LocationManager) getSystemService(LOCATION_SERVICE);  
  
log("\nLocations (starting with last known):");  
Location location  
    = mgr.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);  
// Location location  
//    = mgr.getLastKnownLocation(LocationManager.PASSIVE_PROVIDER);  
// Location location  
//    = mgr.getLastKnownLocation(LocationManager.GPS_PROVIDER);
```

- Significant errors possible
 - Why?

AndroidManifest.xml

- User Permission in manifest

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

- Options: ACCESS_FINE_LOCATION or ACCESS_COARSE_LOCATION
- ACCESS_COARSE_LOCATION for use of NETWORK_PROVIDER using cell-ID and Wi-Fi
- ACCESS_FINE_LOCATION: GPS or NETWORK_PROVIDER

Uses Features

- In addition to request permissions the AndroidManifest.xml file can list features the app uses.
- Google Play uses these tags to filter applications for users
- examples of features: bluetooth, camera, location, network, microphone, nfc (near field communication), sensors, and more!

```
<uses-feature android:name="android.hardware.location.gps"/>  
<uses-feature android:name="android.hardware.location.network"/>
```

Location Manager

- Obtain Location Manager

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    mgr = (LocationManager) getSystemService(LOCATION_SERVICE);
```


Simple Location Program

- Just to demonstrate capabilities
- After setting up listener show all providers
- mgr is LocationManager

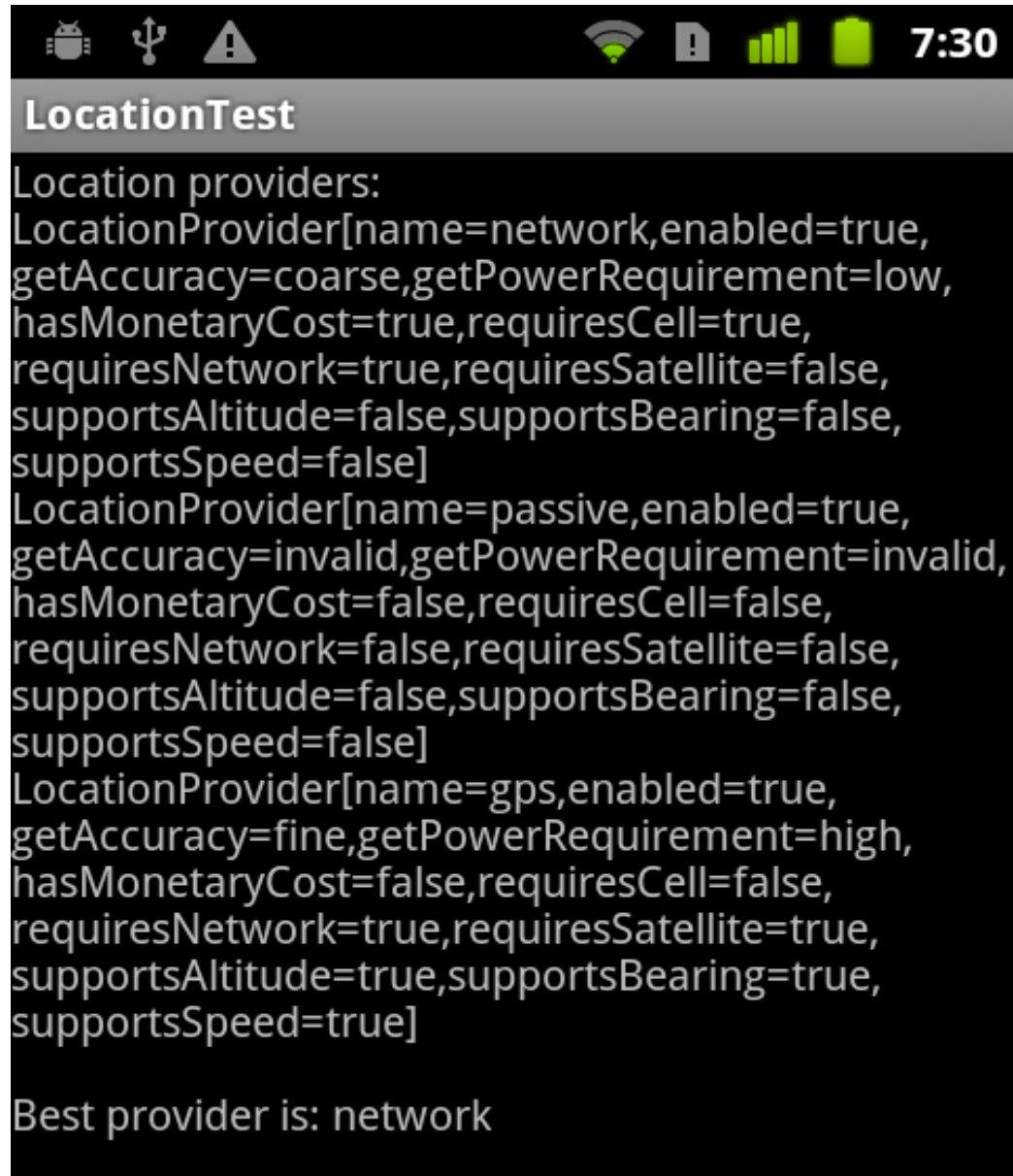
```
/** Write information from all location providers */  
private void dumpProviders() {  
    List<String> providers = mgr.getAllProviders();  
    for (String provider : providers) {  
        dumpProvider(provider);  
    }  
}
```

Properties of Location Providers

- name
- enabled
- accuracy
- power requirements
- monetary cost
- requires cell
- requires network
- requires satellite
- supports altitude
- supports bearing
- supports speed

Program Output

- network (wifi and cell tower id)
- gps
- passive
 - use location updates requested by other applications or services

A screenshot of an Android application interface. At the top, there is a status bar with icons for the Android robot, USB, a warning triangle, Wi-Fi, battery, and the time 7:30. Below the status bar is a grey header bar with the text 'LocationTest'. The main content area has a black background with white text. It lists 'Location providers:' followed by three entries for 'network', 'passive', and 'gps'. Each entry shows various attributes like 'enabled', 'getAccuracy', 'getPowerRequirement', 'hasMonetaryCost', 'requiresCell', 'requiresNetwork', 'requiresSatellite', 'supportsAltitude', 'supportsBearing', and 'supportsSpeed'. At the bottom, it states 'Best provider is: network'.

```
LocationTest

Location providers:
LocationProvider[name=network,enabled=true,
getAccuracy=coarse,getPowerRequirement=low,
hasMonetaryCost=true,requiresCell=true,
requiresNetwork=true,requiresSatellite=false,
supportsAltitude=false,supportsBearing=false,
supportsSpeed=false]
LocationProvider[name=passive,enabled=true,
getAccuracy=invalid,getPowerRequirement=invalid,
hasMonetaryCost=false,requiresCell=false,
requiresNetwork=false,requiresSatellite=false,
supportsAltitude=false,supportsBearing=false,
supportsSpeed=false]
LocationProvider[name=gps,enabled=true,
getAccuracy=fine,getPowerRequirement=high,
hasMonetaryCost=false,requiresCell=false,
requiresNetwork=true,requiresSatellite=true,
supportsAltitude=true,supportsBearing=true,
supportsSpeed=true]

Best provider is: network
```

dev Phones (no cell service)

name	Network	Passive	GPS
enabled	true	true	true
accuracy	coarse	invalid	fine
power req.	low	invalid	high
monetary cost	true??	false	false
request cell	true	false	false
requires network	true	false	true?
requires satellite	false	false	true
supports altitude	false	false	true
supports bearing	false	false	true
supports speed	false	false	true

LocationListener

- Implement class that implements LocationListener interface

Public Methods	
abstract void	<code>onLocationChanged(Location location)</code> Called when the location has changed.
abstract void	<code>onProviderDisabled(String provider)</code> Called when the provider is disabled by the user.
abstract void	<code>onProviderEnabled(String provider)</code> Called when the provider is enabled by the user.
abstract void	<code>onStatusChanged(String provider, int status, Bundle extras)</code> Called when the provider status changes.

Obtaining Locations

- Register the `LocationListener` to receive location updates
- `locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 15000, 10, locationListener);`
 - provider: name of provider to register with
 - minTime: the minimum time interval for notifications, in milliseconds. only a hint to conserve power, and actual time between location updates may be greater or lesser than this value.
 - minDistance: min distance interval for notifications in meters
 - the listener itself

requestLocationUpdates

- More on arguments
- 0 for minTime AND minDistance indicate obtain updates as frequently as possible
- for *background services* recommended minTime \geq 300,000 ms to avoid consuming too much power with the GPS or Wi-Fi receivers
- 300,000 ms = 5 minutes
- clearly less for apps in the foreground

Location Listener

```
// Acquire a reference to the system Location Manager
LocationManager locationManager = (LocationManager) this.getSystemService(Context.LOCATION_SERVICE)

// Define a listener that responds to location updates
LocationListener locationManager = new LocationListener() {
    public void onLocationChanged(Location location) {
        // Called when a new location is found by the network location provider.
        makeUseOfNewLocation(location);
    }

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

    public void onProviderEnabled(String provider) {}

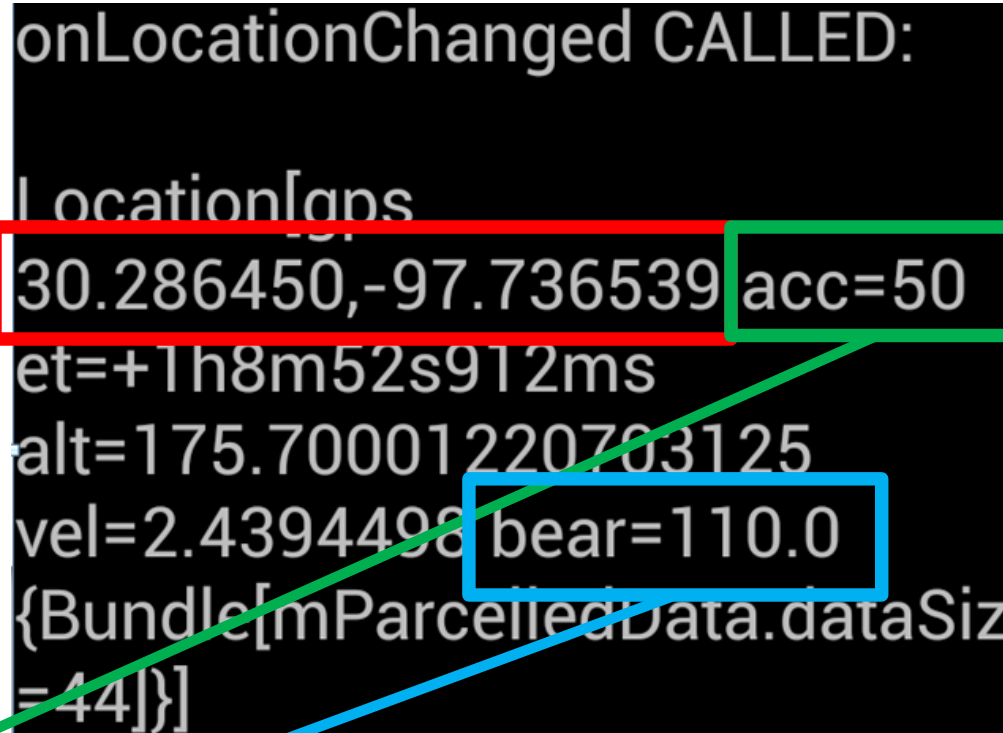
    public void onProviderDisabled(String provider) {}
};

// Register the listener with the Location Manager to receive location updates
locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 0, 0, locationManager);
```

Location Data

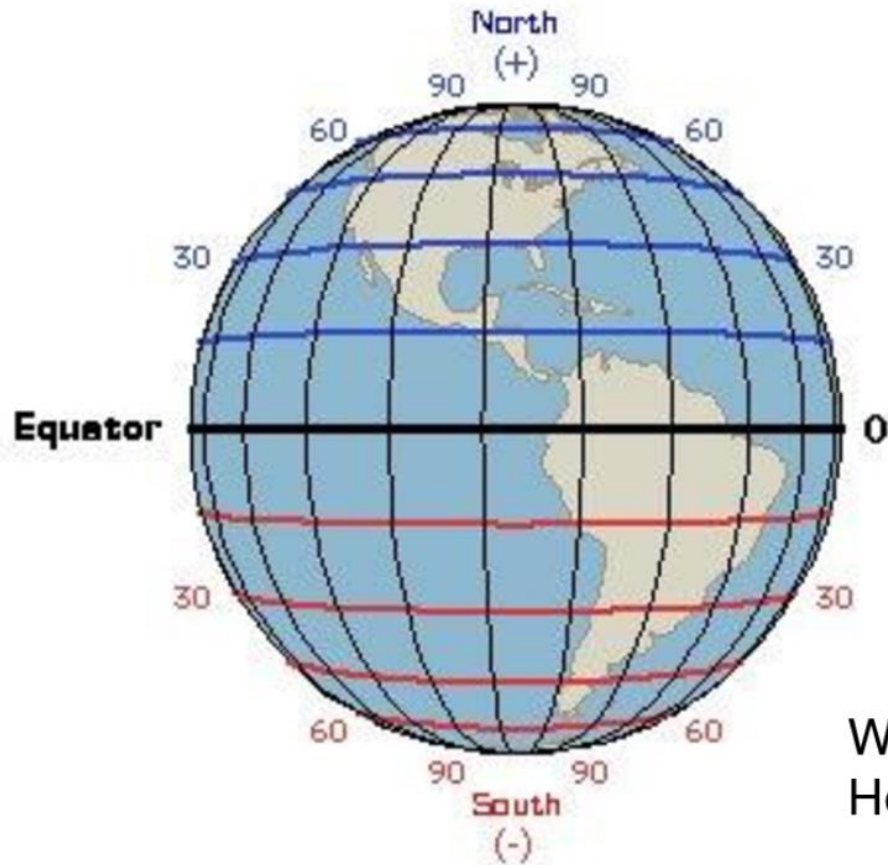
- onLocationChange method in the LocationListener receives Location objects
- toString shown
- latitude, longitude, estimated accuracy in meters, bearing

```
onLocationChanged CALLED:  
Location[gps  
30.286450,-97.736539 acc=50  
et=+1h8m52s912ms  
alt=175.70001220703125  
vel=2.4394498 bear=110.0  
{Bundle[mParcelledData.dataSize  
=44]}]
```



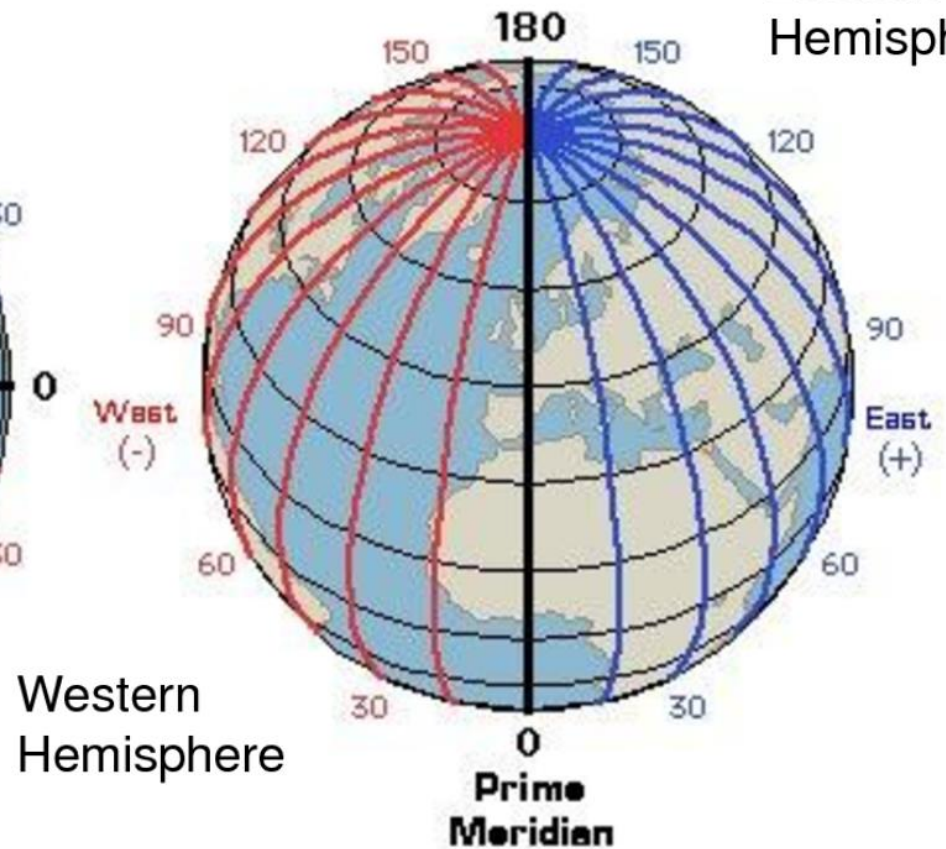
Latitude and Longitude

Northern Hemisphere



Southern Hemisphere

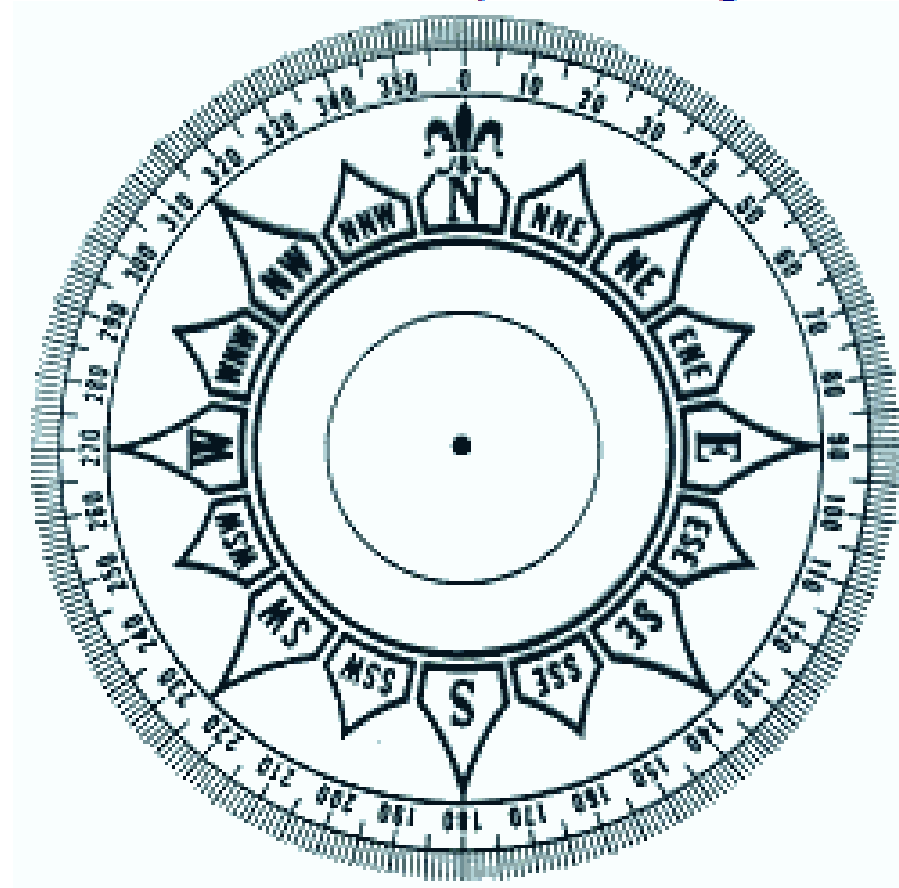
Eastern Hemisphere



Western Hemisphere

Bearing

- direction
- 360 degrees
- degrees east of north
- 0 = north
- 90 = east
- 180 = south
- 270 = west



Sample GPS Locations

- **ET = Elapsed time:**
Time passed since device start up
- **Altitude in meters**
- **units for velocity:**
meters / sec
– 1 m/s \approx 2.2 mph

```
onLocationChanged CALLED:  
Location[gps  
30.286450,-97.736539 acc=50  
et=+1h8m52s912ms  
alt=175.70001220703125  
vel=2.4394498 bear=110.0  
{Bundle[mParcelledData.dataSize  
=44]}}
```


toString for Location

onLocationChanged CALLED:

Location[gps

30.286450,-97.736539 acc=50

et=+1h8m52s912ms

alt=175.70001220703125

vel=2.4394498 bear=110.0

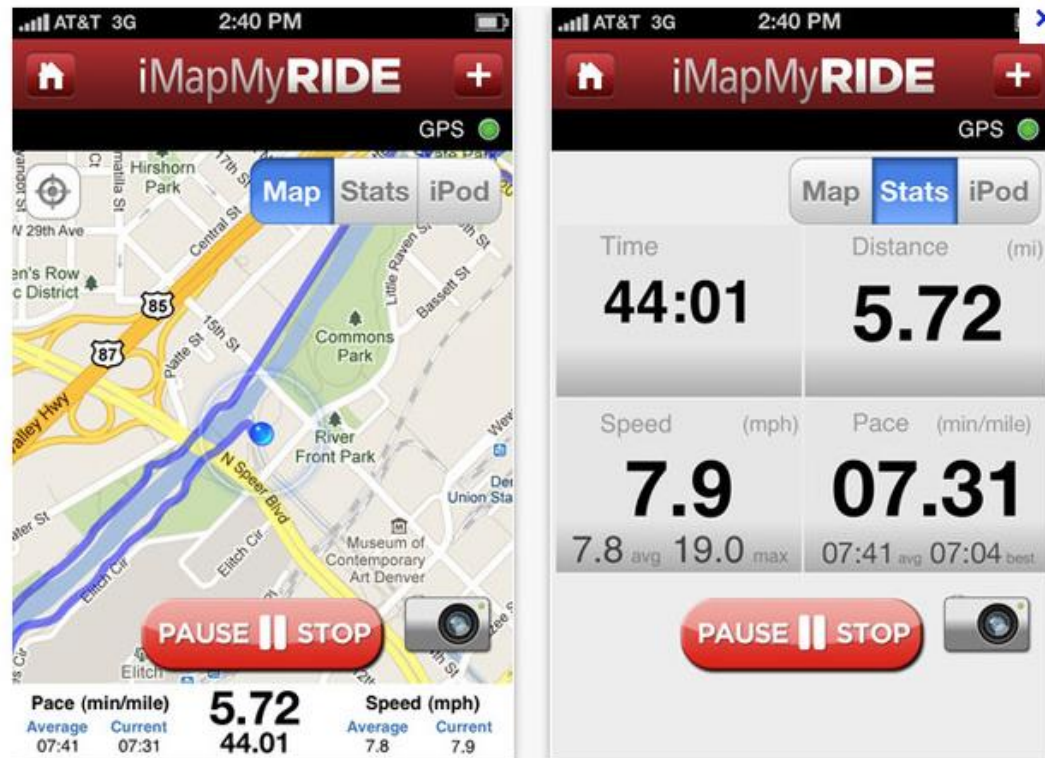
{Bundle[mParcelledData.dataSize
=44]}}

Location Strategies

- Location aware applications
 - compelling? better information to user?
- GPS -> slow, only works outdoors, consumes lots of power, very accurate
- Network -> fast, works indoor and outdoor, uses less power, less accurate
- Issues: multiple sources (cell id with call plan, wifi, gps), user movement, accuracy of locations

Getting a Fix

- Some applications (driving directions, sport tracking) require constant location data
 - using battery is expected



Periodic Location Updates

- Many location aware applications do not need a constant stream of location updates
- Obtaining location pattern:
 1. Start application.
 2. Sometime later, start listening for updates from desired location providers.
 3. Maintain a "current best estimate" of location by filtering out new, but less accurate fixes.
 4. Stop listening for location updates.
 5. Take advantage of the last best location estimate.

Getting Location

- Timeline for getting location based on pattern described:

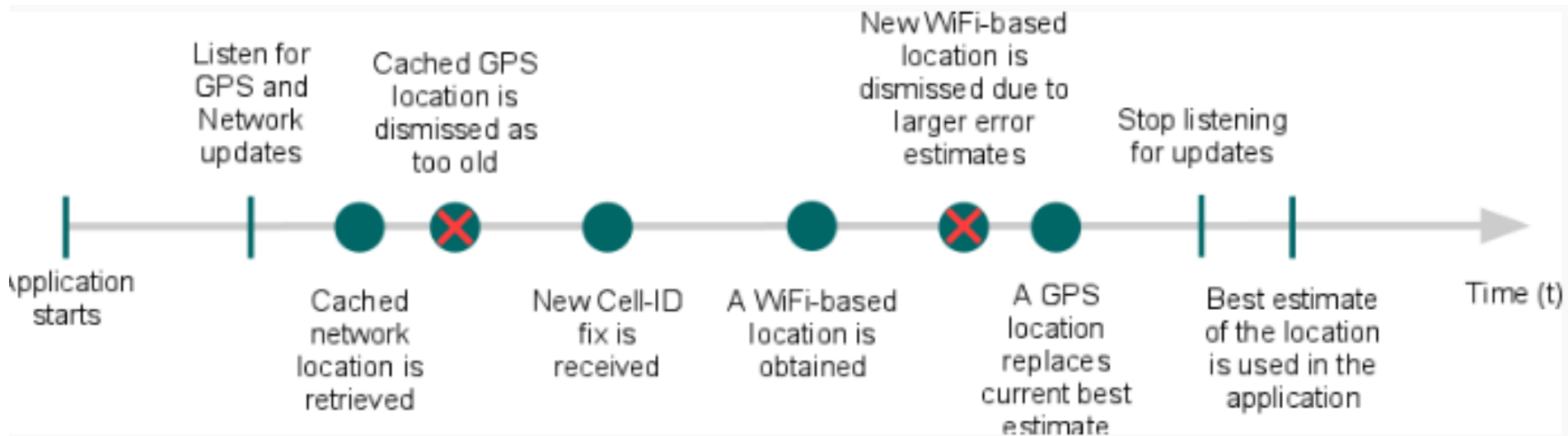


Figure 1. A timeline representing the window in which an application listens for location updates.

Last Known Location

- Recall, application is part of a larger system
- other applications may have asked for location and we can use those locations via the LocationManager

```
LocationProvider locationProvider = LocationManager.NETWORK_PROVIDER;  
// Or use LocationManager.GPS_PROVIDER  
  
Location lastKnownLocation = locationManager.getLastKnownLocation(locationProvider);
```


Current Best Estimate

- The most recent location, may not be the most accurate
- Evaluating a location
 - how long has it been since the current best estimate?
 - is the accuracy of the new location update better than the best estimate?
 - what is the source of the location? which do you trust more?

LocationManager - Useful Methods

- `addProximityAlert(double latitude, double longitude, float radius, long expiration, PendingIntent intent)`
 - Sets a proximity alert for the location given by the position (latitude, longitude) and the given radius.
- `List<String> getAllProviders()`
 - Returns a list of the names of all known location providers.
- `Location getLastKnownLocation(String provider)`
 - Returns a Location indicating the data from the last known location fix obtained from the given provider.
- `Location class: float distanceTo(Location dest)`
 - Returns the approximate distance in meters between this location and the given location.

Sample Proximity Alert

```
public void setNearNotice(View v) {  
    double gdcLat = Double.parseDouble("30.286263");  
    double gdcLong = Double.parseDouble(getString(R.string.gdc_long));  
    Intent showLeavingGDCIntent = new Intent(this, LeavingGDC.class);  
    int requestCode = 2008;  
    PendingIntent showLeavingGDCPendingIntent  
        = PendingIntent.getActivity(this,  
            requestCode, showLeavingGDCIntent,  
            PendingIntent.FLAG_UPDATE_CURRENT);  
    if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)  
        && ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_BACKGROUND_LOCATION))  
        mgr.addProximityAlert(gdcLat, gdcLong,  
            100, -1, showLeavingGDCPendingIntent);  
}  
}
```

GEOCODING

Geocoding

- LocationManger and GooglePlayServices provide a Location object
- Contains latitude and longitude
- Latitude, degrees north of south of equator North Pole = 90N or +90, equator = 0, South Pole = 90S or -90
- longitude, degrees east or west of prime meridian, 0 to -180 to west, 0 to +180 to west

Geocoding

- Various databases exists to give address(es) at a given lat and long
- Web Service?
- GeoCoder class in Android provides:
 - geocoding: convert physical address to lat / long
 - reverse geocoding: convert lat / long to physical address

Geocoding

- Accessing Google database of addresses
- Making network calls may block the Activity
- Key to not "BLOCK THE UI THREAD"
- Meaning, don't do expensive computations or possibly slow operations in the code for an Activity
- Another use of AsyncTask

Geocoding

- When address button clicked use a Geocode to try and get address

// address Button clicked, show address

```
public void showAddress(View v) {  
    tryCount = 0;  
    getAddress();  
}
```

```
private void getAddress() {  
    AsyncTask<Geocoder, Void, List<Address>>  
        addressFetcher = new AddFetch();  
    Geocoder gc = new Geocoder(this, Locale.US);  
    addressFetcher.execute(gc);  
}
```


Reverse Geocoding

- Get lat and long from last known location of LocationTest app

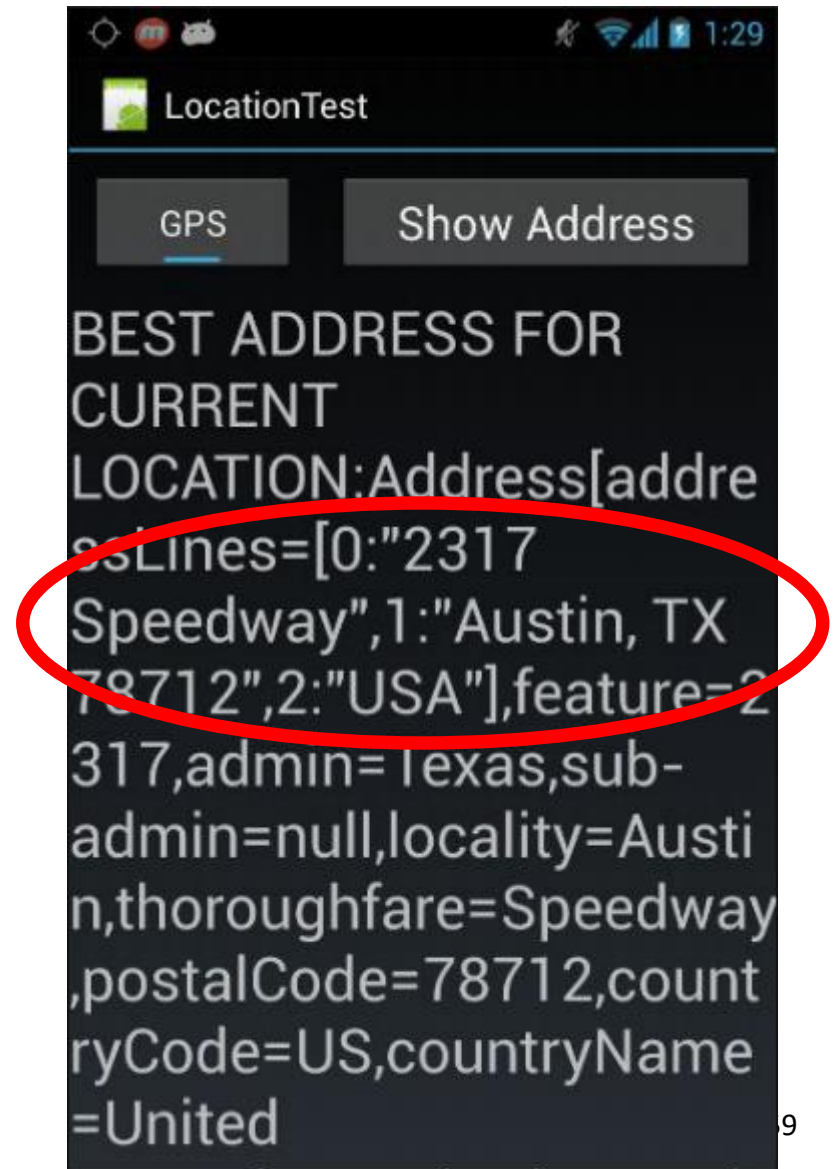
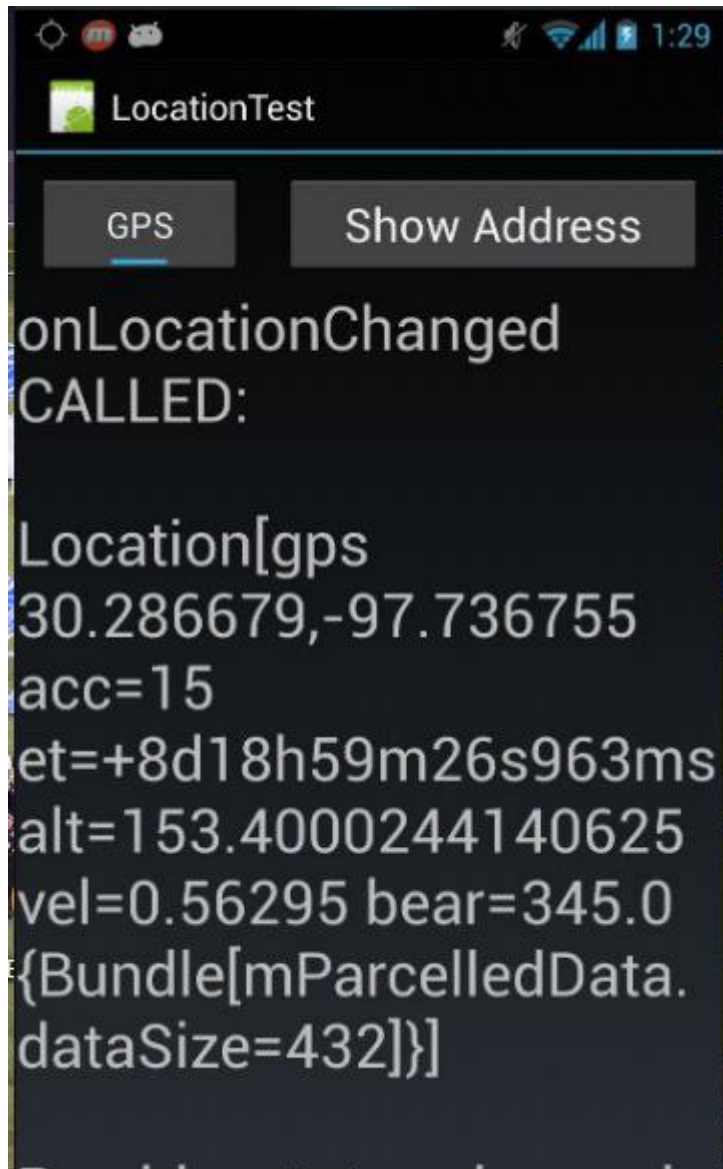
```
private void tryReverseGeocoding(Geocoder gc) {  
    if (lastKnownLocation != null) {  
        double lat = lastKnownLocation.getLatitude();  
        double lng = lastKnownLocation.getLongitude();  
        Log.d(TAG, "REVERSE GEO CODE TEST lat: " + lat);  
        Log.d(TAG, "REVERSE GEO CODE TEST long: " + lng);  
        addresses = null;  
        try {  
            addresses = gc.getFromLocation(lat, lng, 10); // maxR  
        } catch (IOException e) {  
        }  
    } else {  
        output.append("No locations yet. Please try later.\n\n");  
    }  
}
```

If Successful

```
protected void onPostExecute(List<Address> result) {  
    if(result == null) {  
        tryAgain();  
        Log.d(TAG, "No addresses from Geocoder. Trying again. " +  
            "Try count: " + tryCount);  
    } else {  
        output.append("Number of addresses " +  
            "at current location :" + addresses.size());  
        output.append("BEST ADDRESS FOR CURRENT LOCATION:\n");  
        output.append(addresses.get(0).toString());  
        Log.d(TAG, "reverse geocoding, " +  
            "addresses from lat and long: "  
            + addresses + " " + addresses.size());  
    }  
}
```

- Geocoder returns a list of possible addresses

Demo

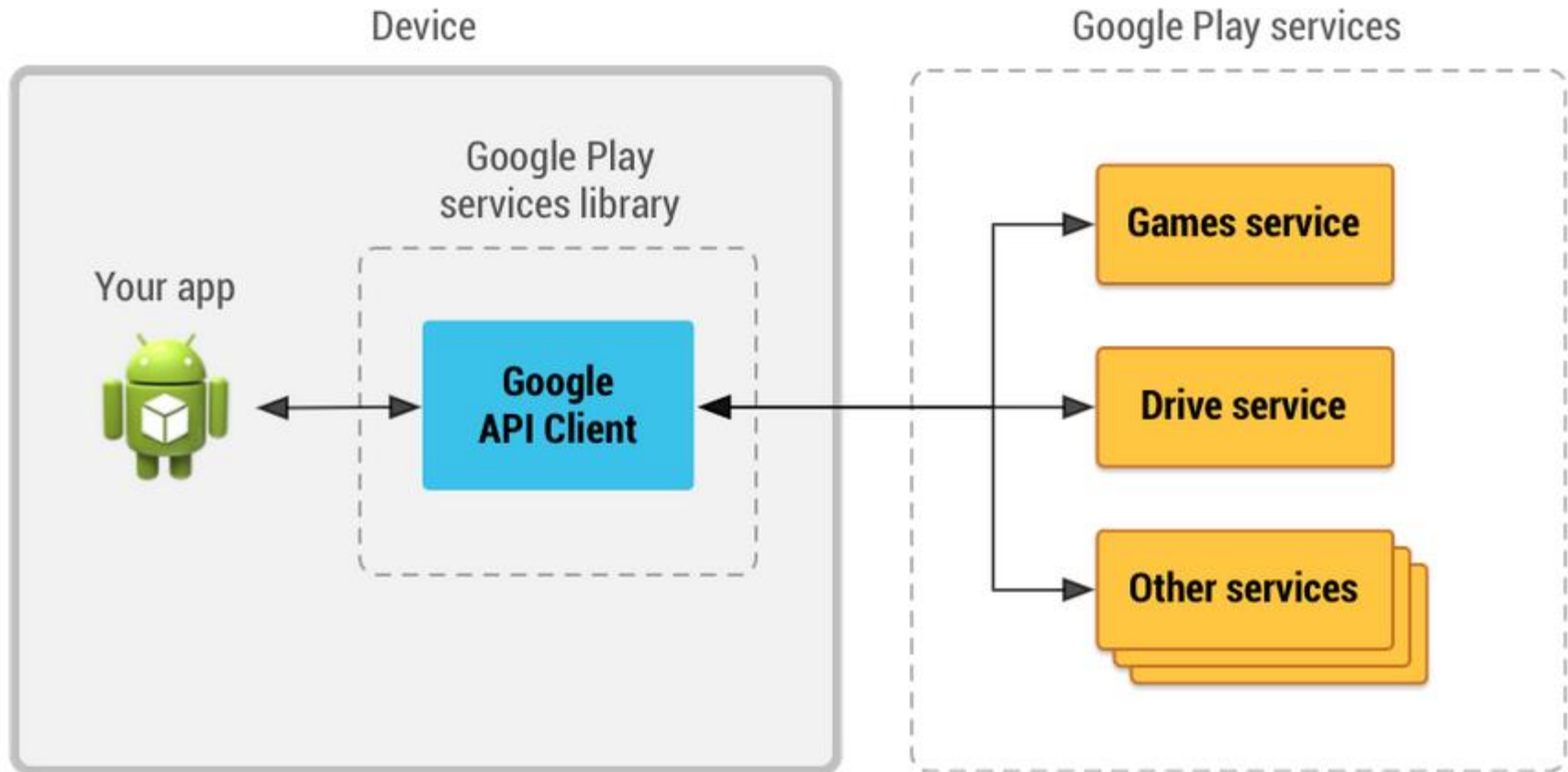


LOCATION SERVICES VIA GOOGLE PLAY SERVICES

Alternative?

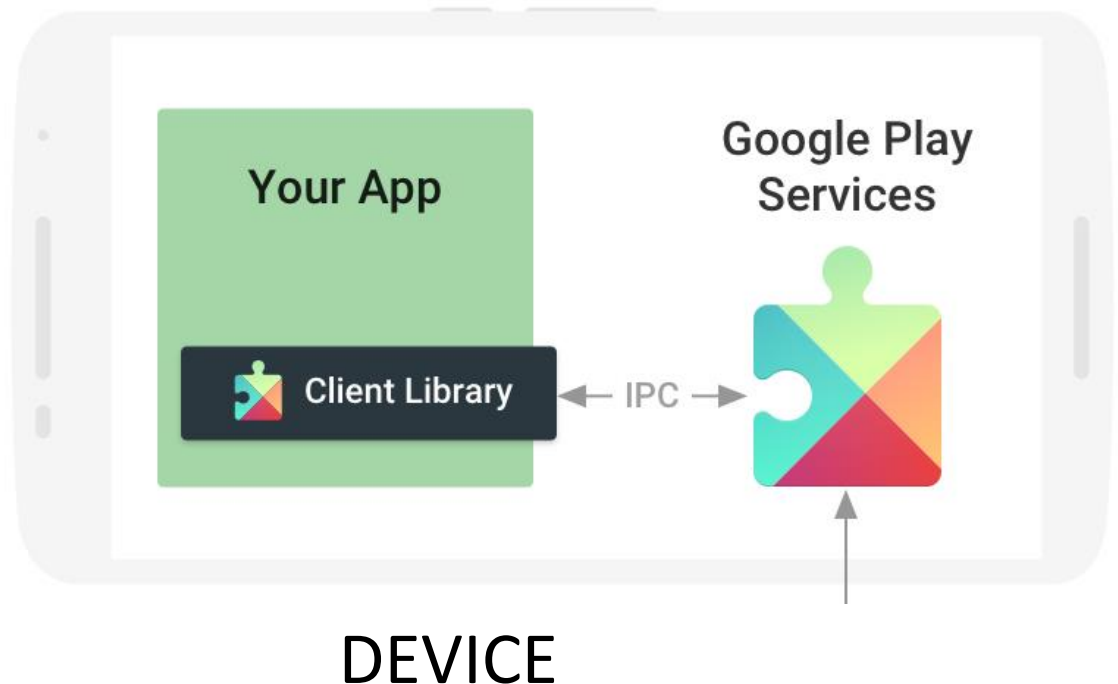
- Google Location Services API
 - "part of Google Play Services, provides a more powerful, high-level framework that automatically handles location providers, user movement, and location accuracy. It also handles location update scheduling based on power consumption parameters you provide. In most cases, you'll get better battery performance, as well as more appropriate accuracy, by using the Location Services API. "

Google Play Services



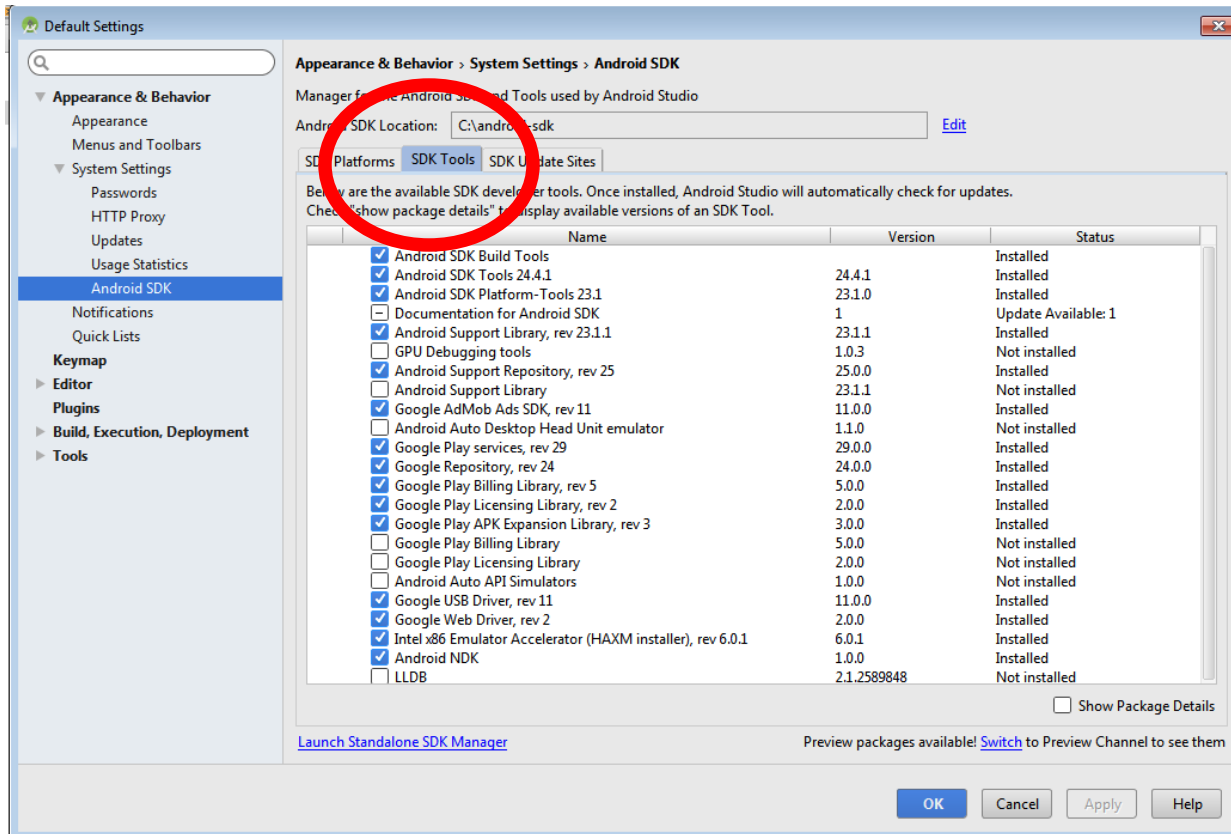
Google Play Services

- Extra APIs provided by Google
- Allows your app to access various Google services such as:
 - maps
 - game services
 - **location APIs**
 - ads
 - OAuth
 - many others



Using Google Play Services

- Download Google Play Services for Android SDK via SDK Manager



☐ Android Auto Desktop Head Unit emulator

1.1.0

☒ Google Play services, rev 29

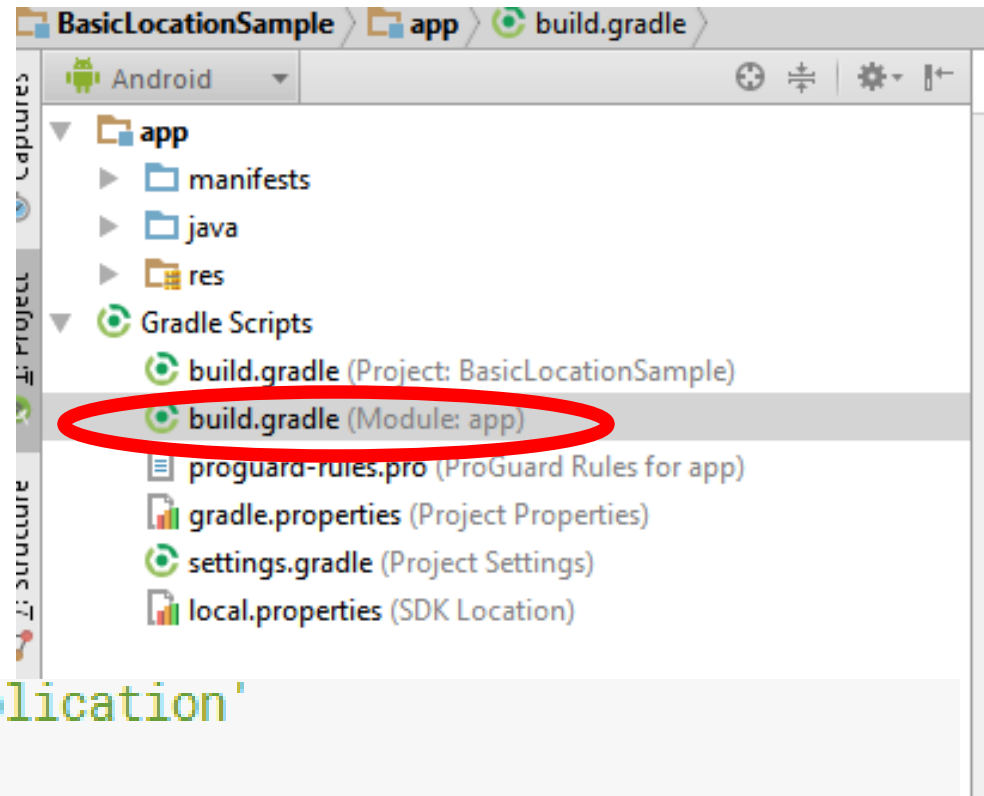
29.0.0

☒ Google Repository, rev 24

24.0.0

Setting up Google Play Services

- In target app must add dependency in the Gradle build file for the app / module - not the project app



```
apply plugin: 'com.android.application'
```

```
...
```

```
dependencies {  
    compile 'com.google.android.gms:play-services:8.4.0'  
}
```

Getting Location

- Must still request permission for location access in manifest file
- Create FusedLocationProviderClient in onCreate method of Activity that wants location

```
private FusedLocationProviderClient mFusedLocationClient;

// ..

@Override
protected void onCreate(Bundle savedInstanceState) {
    // ...

    mFusedLocationClient = LocationServices.getFusedLocationProviderClient(this);
}
```

Getting Location

- When Location desired:

```
mFusedLocationClient.getLastLocation()
    .addOnSuccessListener(this, new OnSuccessListener<Location>() {
        @Override
        public void onSuccess(Location location) {
            // Got last known location. In some rare situations this can
            if (location != null) {
                // Logic to handle location object
            }
        }
    });
```

Geofencing

- Location Services also supports creating and monitoring geofences
- App can create a geo fence
 - latitude and longitude
 - radius
- Once a geofence is created app can ask Location Services for notifications
 - when device enters geofence
 - when device leaves geofence
 - when device has been present (dwells) in geofence for a specified amount of time

Geofencing

