CS378 - 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
  • Android can use GPS and the Network Location Provider which combines cell-ID and Wi-Fi data
Location, Location, Location

- Dead reckoning
- radar fix
- visual fix
- Loran
- Omega
- Navsat
- GPS
- Active Sonar
- Inertial Navigation System
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
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 also have an effect
GPS Accuracy

Horizontal Position Error Histogram: 1 January - 31 March 2011

<= 95% Horizontal Error (2.199 m)
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

• Obtaining User Location
• GPS
  – most accurate but,
  – only works OUTDOORS
  – quickly consumes battery power
  – delay in acquiring satellites or re-acquiring if lost
How does Wi-Fi Fix Location?

• Use to use StreetView cars
• Now, use the devices themselves to map locations to wi-fi spots
• Apple and Microsoft do the same thing
• default on dev phones was checked
Google Location Services

My Location

Use wireless networks
See location in applications (such as Maps) using wireless networks

Location consent
Allow Google's location service to collect anonymous location data. Collection will occur even when no applications are running!

Agree
Disagree
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
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

```java
@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
Bearing

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

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

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)

<table>
<thead>
<tr>
<th>name</th>
<th>Network</th>
<th>Passive</th>
<th>GPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabled</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>accuracy</td>
<td>coarse</td>
<td>invalid</td>
<td>fine</td>
</tr>
<tr>
<td>power req.</td>
<td>low</td>
<td>invalid</td>
<td>high</td>
</tr>
<tr>
<td>monetary cost</td>
<td>true??</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>request cell</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>requires network</td>
<td>true</td>
<td>false</td>
<td>true?</td>
</tr>
<tr>
<td>requires satellite</td>
<td>false</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>supports altitude</td>
<td>false</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>supports bearing</td>
<td>false</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>supports speed</td>
<td>false</td>
<td>false</td>
<td>true</td>
</tr>
</tbody>
</table>
LocationListener

- Implement class that implements LocationListener interface

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abstract void</td>
<td>onLocationChanged (Location location)</td>
<td>Called when the location has changed.</td>
</tr>
<tr>
<td>abstract void</td>
<td>onProviderDisabled (String provider)</td>
<td>Called when the provider is disabled by the user.</td>
</tr>
<tr>
<td>abstract void</td>
<td>onProviderEnabled (String provider)</td>
<td>Called when the provider is enabled by the user.</td>
</tr>
<tr>
<td>abstract void</td>
<td>onStatusChanged (String provider, int status, Bundle extras)</td>
<td>Called when the provider status changes.</td>
</tr>
</tbody>
</table>
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
// 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 locationListener = 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, locationListener);
Location Data

- `onLocationChange` method in the `LocationListener` receives `Location` objects
- `toString` shown
- `latitude`, `longitude`, `timestamp`, possibly `altitude`, `speed`, and `bearing`
Sample GPS Locations

- units for altitude and accuracy: meters
- units for speed: meters / sec
- units for time: UTC (coordinated universal time) time of this fix, in milliseconds since January 1, 1970.
- Java Date class has constructor that accepts UTC
Sample GPS Locations

Location[mProvider=gps,
mTime=1333414860000,mLatitude=30.51601886364164,mLongitude=-97.71347013064596,mHasAltitude=true,mAltitude=210.0,mHasSpeed=true,
mSpeed=0.29998752,mHasBearing=true,
mBearing=315.55438,mHasAccuracy=true,
mAccuracy=20.0,mExtras=Bundle[mParcelledData.
dataSize=4]]
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, 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.

http://developer.android.com/guide/topics/location,strategies.html
Getting Location

• Timeline for getting location based on pattern described:

![Diagram](Image)

*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

```java
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.
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."