Accelerating Information Technology Innovation

http://aiti.mit.edu

India Summer 2012

Lecture 10 – Location Data and GPS

A Problem

- Ashish and Kalpana are running errands separately, but want to meet for lunch!
- Ashish is waiting for a suit to be tailored, while Kalpana is meeting her hairstylist later.
- How can an app help them find a place to meet in the middle?

"Indian cuisine" by Kirti Poddar
Licensed under a Creative Commons
Attribution-Share Alike 2.0 Generic License
<http://www.flickr.com/photos/feastguru_kirti/2242523634/>

Location Services on Android

- Gets location fix from GPS, Mobile and Wifi Networks, or both.
- Generally starts course, becomes finer.
- Can be used to with Google Maps API to display maps on the phone.

Using Location Services

- Connect to `LocationManager` Service.
- Set a `LocationListener` on the manager.
- Ask the `LocationManager` to provide location updates.
- When done, stop listening for updates.

Connecting to the `LocationManager`

```
// Connect to the system location service.
// “this” should be a reference to the current Activity
LocationManager locationManager =
  (LocationManager) this.getSystemService(Context.LOCATION_SERVICE);
```

Setting the LocationListener

```
// Defines the location listener.
LocationListener locationListener = new LocationListener() {
    public void onLocationChanged(Location location) {
        // Called when the location provider finds a new location.
    }
    public void onProviderDisabled(String provider) {
        // Called when the user disables the location provider.
    }
    public void onProviderEnabled(String provider) {
        // Called when the user enables the location provider.
    }
    public void onStatusChanged(String provider, int status,
        Bundle extras) {
        // Called when the location provider availability changes.
    }
}
```

Setting the LocationListener

```
// Defines the location listener.
LocationListener locationListener = new LocationListener() {
    public void onLocationChanged(Location location) {
        // Called when the location provider finds a new location.
    }
    public void onProviderDisabled(String provider) {
        // Called when the user disables the location provider.
    }
    public void onProviderEnabled(String provider) {
        // Called when the user enables the location provider.
    }
    public void onStatusChanged(String provider, int status,
        Bundle extras) {
        // Called when the location provider availability changes.
    }
}
```

Setting the `LocationListener`

- `location.getAccuracy()`/`.hasAccuracy()`:
  accuracy of location in meters.
- `location.getBearing()`/`.hasBearing()`:
  bearing (movement direction) in degrees
  east of north.
- `location.getLatitude()`/`.getLongitude()`:
  latitude/longitude in degrees (North, East positive)
- `location.getSpeed()`/`.hasSpeed()`:
  speed of movement.

Setting the `LocationListener`

- `location.getTime()`:
  time location was measured (seconds since 1970).
- `location.distanceTo(anotherLocation)`:
  meters between location and anotherLocation.
- `location.bearingTo(anotherLocation)`:
  direction to anotherLocation from location
  (in degrees east of north).

Setting the `LocationListener`

- Not every update is a good one!
- Use some "heuristics" to determine if a location is better than another (e.g. source, better accuracy, time (`location.getTime()`, etc.)
- Listen to the location over time (but not too long!)

Ask the LocationManager for updates

```
// Connect to the system location service.
// “this” should be a reference to the current Activity
locationManager.requestLocationUpdates(
    // Where to read the location.  Can be either NETWORK_PROVIDER
    // to read from Mobile/Wifi networks, or GPS_PROVIDER to read
    // from GPS.  The latter is more accurate, but more sensitive.
    LocationManager.NETWORK_PROVIDER,
    // Minimum time between updates in milliseconds it is good to
    // keep this as long as possible to save battery life.
    0,
    0,  // Minimum distance between updates in meters.
    locationListener  // Your location listener
);
```

Stop Listening to Updates

```
// Stop listening to the system location service.
locationManager.removeUpdates(locationListener);
```

Getting Location Quickly

- It may take a while to get the first update.
- Call the following to get a (possibly outdated) cached location:

```
// Get the last (cached) location.
Location lastKnownLocation =
    locationManager.getLastKnownLocation(locationProvider);
```

Permissions

- Just as your app needs permissions to access the internet on a phone (android.permission.INTERNET), so do location services.
  - android.permission.ACCESS_COURSE_LOCATION grands access to `LocationManager.NETWORK_PROVIDER`
  - android.permission.ACCESS_FINE_LOCATION also grands access to `LocationManager.GPS_PROVIDER`

Adding Location Services

Adding Location Services

Create a new project and add two buttons in addition to the "Hello World" TextView: "Listen" and "Stop Listening"

Adding Location Services

In the `onCreate()` *method, create a* `LocationListener` *and save it as an instance variable.*

When it receives an update, it should set the `TextView` *to read "Latitude ##, Longitude ##" where ## is filled in with the latitude and longitude of the* `Location`.

Adding Location Services

When the "Listen" button is clicked, start listening for location updates.

Also, set the `TextView` *text to the most recent location.*

Adding Location Services

When the "Stop Listening" button is clicked, stop listening for location updates.

Also, set the `TextView` *text to "Not Listening."*

Testing Location Services

Two Methods

- Connect to the emulator console
- Send coordinates (as GPX/KML) via DDMS

The Emulator Console

- Start your app.
- Note the port number of your emulator (e.g. :5554, etc.)
- Open a command-line.
- Type “telnet localhost [port-number]”
- Send fixes using
  “geo fix [latitude] [longitude] [altitude]”

Using DDMS

- Start your app.
- Open the DDMS perspective ("Window > Open Perspective > Other... > DDMS")
- Upload a GPX or KML file in the "Emulator Control" tab.
  - GPX is a format for GPS tracks (OSMTracker)
  - KML is a format to store places (Google Earth)

An Exercise

Exercise: Testing Location Services

Choose an appropriate method to test with and try to input the location "19.131637°N 72.915713°E"

References

- Android Developer Site: "Location Strategies" <http://developer.android.com/guide/topics/location/strategies.html>

- A discussion of why you might want to be careful about your location data: "FourSquare and security: I know where you live..." <http://econsultancy.com/us/blog/6423-foursquare-and-security-issues-i-know-where-you-live-2>

- Wikipedia: "GPS eXchange Format" <http://en.wikipedia.org/wiki/GPS_eXchange_Format>

- Wikipedia: "Keyhole Markup Language" <http://en.wikipedia.org/wiki/Keyhole_Markup_Language>