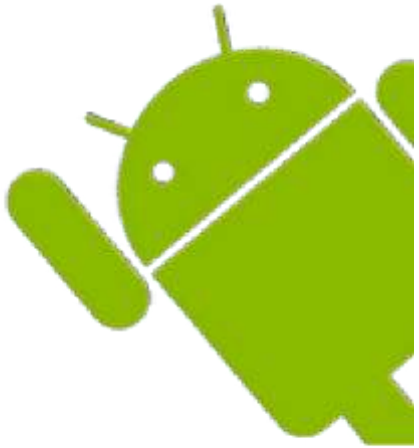
Mobile app development – week 12



Location aware apps

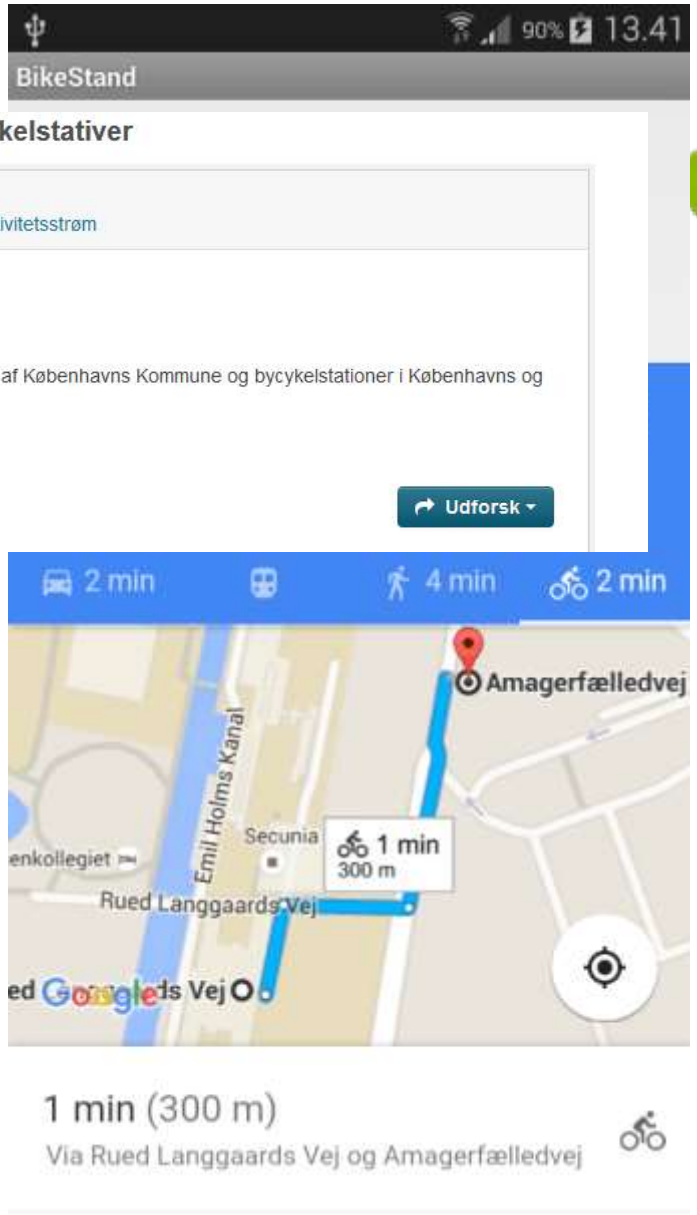
Android components
- Broadcast receiver

Course evaluation

No exercises

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

Location aware apps



Geolocation basics

Coordinates (lat, long) e.g.

55.83, 12.43

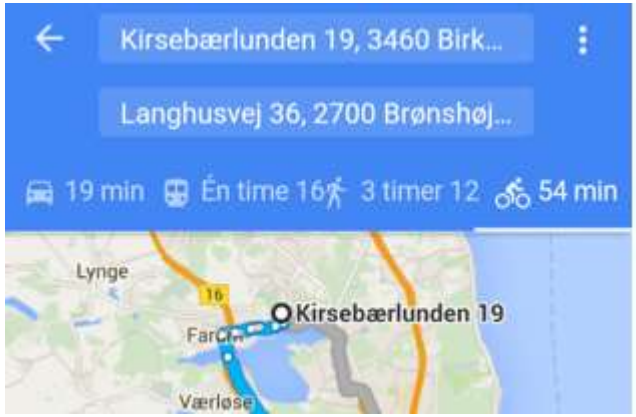
Class Location:
http://developer.android.com/reference/android/location/Location.html

REST interface for Google maps

http://maps.google.com?q=55.83, 12.43

q=
is used to specify the search query in Google maps search, eg :
http://maps.google.com?q=newyork or
http://maps.google.com?q=51.03841,-114.01679

t=
Sets the kind of map shown. Can be set to:
m – normal map,k – satellite,h – hybrid,p – terrain



saddr=
Sets the starting point for directions searches.

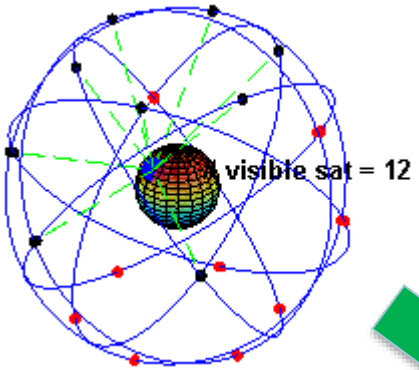
daddr=
Sets the end point for directions searches

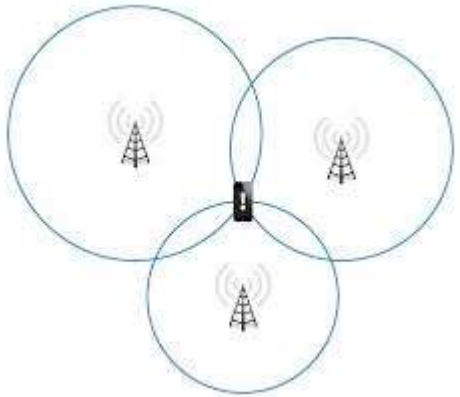
e.g. http://maps.google.com?saddr=*sLat,sLon*&daddr=*dLat,dLon*

dirflg=b

http://moz.com/ugc/everything-you-never-wanted-to-know-about-google-maps-parameters

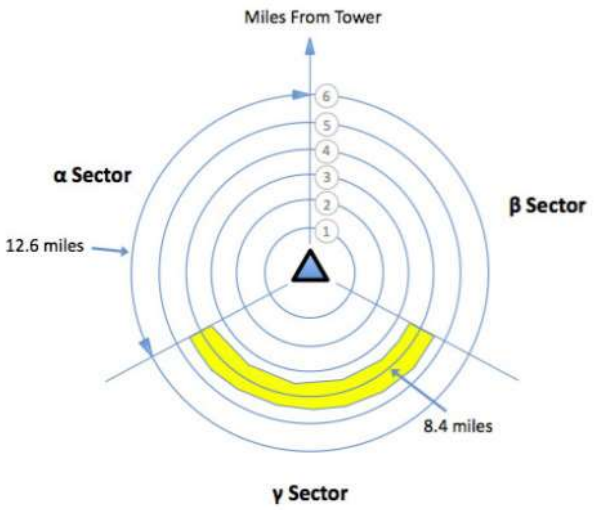
Finding your coordinates: GPS

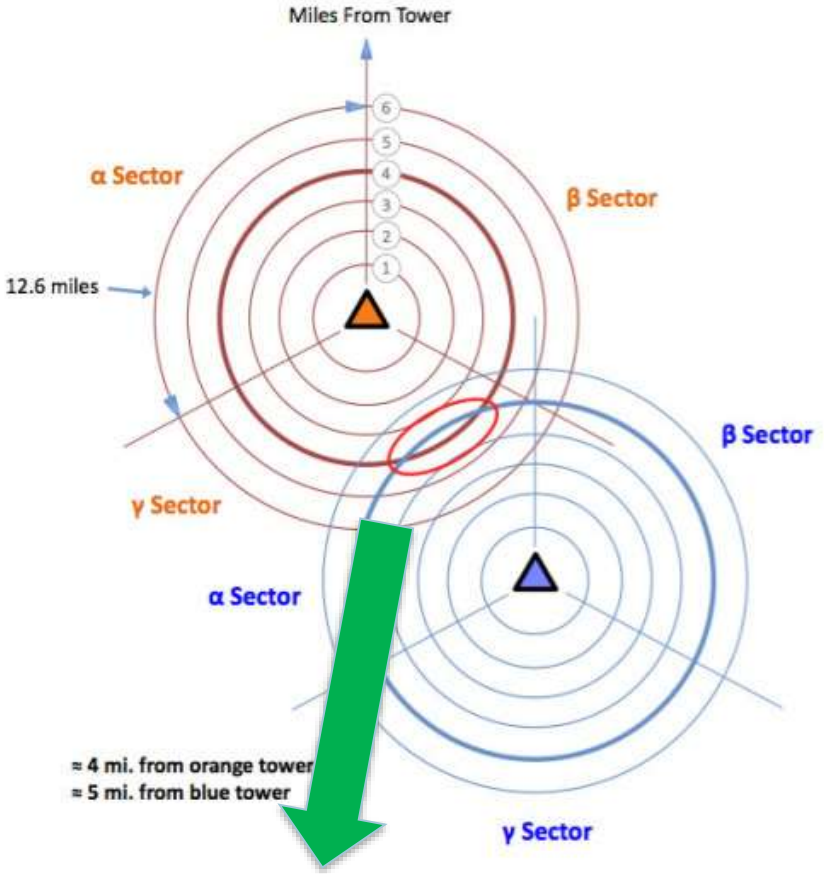




55.83, 12.43

Finding your coordinates: cell towers

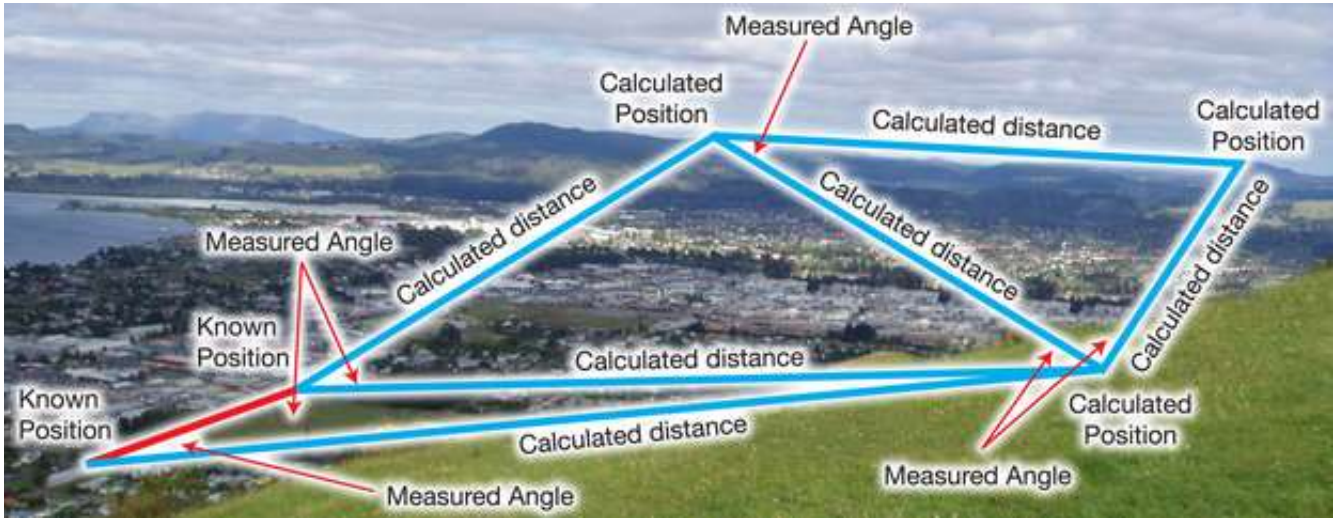




55.83, 12.43

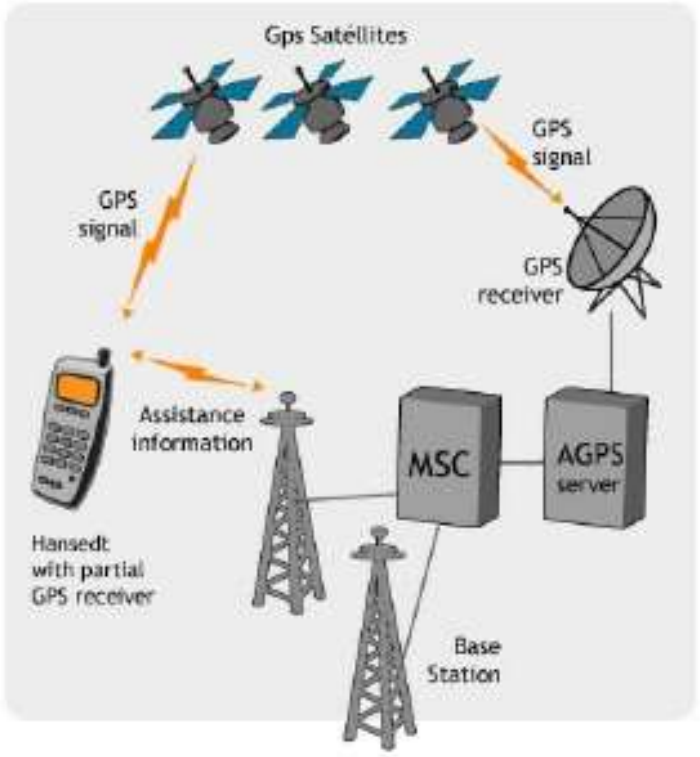
Triangulation vs. trilateration

Triangulation:



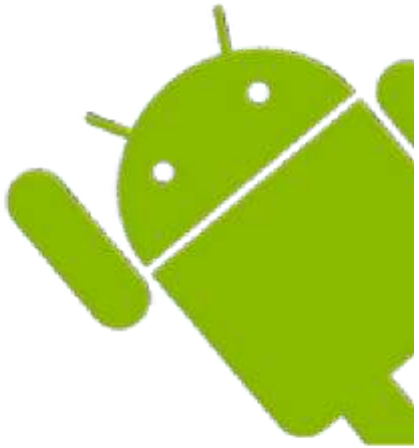
Trilateration:

Assisted GPS and indoor location



Bluetooth, Wifi, NFC, Barcodes, …

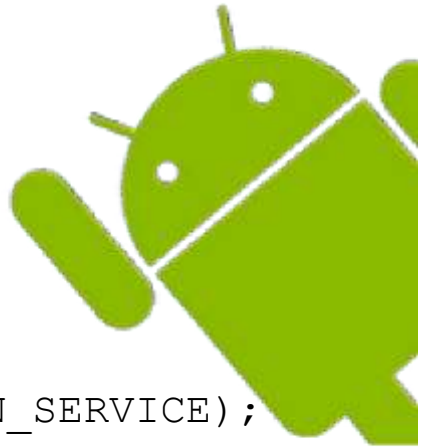
Getting the location on a smart phone



Much more than just GPS !!!

	GPS	AGPS	Cell towers	Wi-Fi	BLE	NFC/ Barcode
Precision	High	High	Low	Medium	High	Very high
Speed	Low	Medium	High	High	High	High
Power usage	High	Medium	Low	Medium	Low	Very low

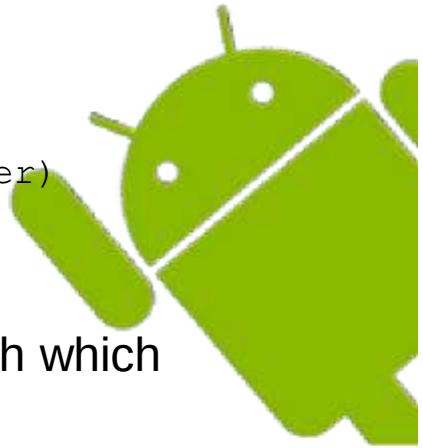
Android LocationManager



```
public class  "component" implements LocationListener {
 private LocationManager l;
 @Override
 protected void onCreate(Bundle savedInstanceState) {
   …
   l= (LocationManager) getSystemService(Context.LOCATION_SERVICE);

   l.requestLocationUpdates( …  );
  }
  // LocationListener interface
  @Override
  public void onLocationChanged(Location location) {
    // Called when a new location is found
    makeUseOfNewLocation(location);
  }

  @Override
  public void onProviderDisabled(String provider) { }
  @Override
  public void onProviderEnabled(String provider) {  }
  @Override
  public void onStatusChanged(String prov, int stat, Bundle ext){ }
}
```

LocationUpdates

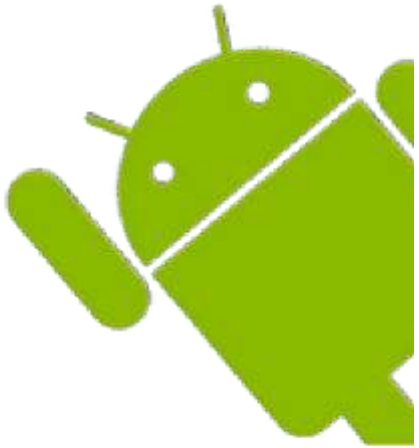
```
l.requestLocationUpdates(provider, minTime, minDist, listener)
```



provider	the name of the provider with which we would like to regiser.
minTime	minimum time interval between location updates (in milliseconds).
minDistance	minimum distance between location updates (in meters).
listener	a LocationListener whose onLocationChanged(Location) method will be called for each location update.

```
l.requestLocationUpdates(LocationManager.GPS_PROVIDER, 2000, 100, this);
l.requestLocationUpdates(LocationManager.NETWORK_PROVIDER,2000,100,this);
```

Permissions



In order to use location services, you need to request the proper permissions from the user, e.g.:

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

- ACCESS_FINE_LOCATION for GPS/AGPS
- ACCESS_COARSE_LOCATION for cell towers and wi-fi

By asking for fine location permission, you automatically receive for coarse as well.

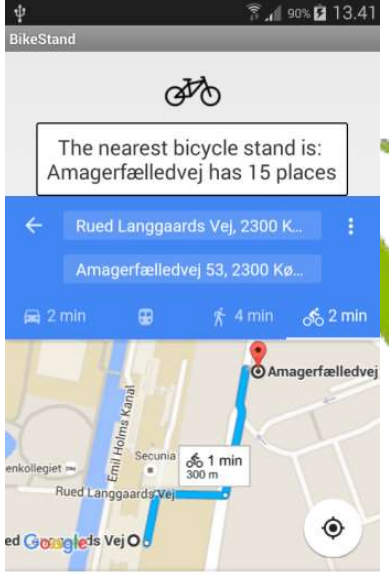
BikeStand app

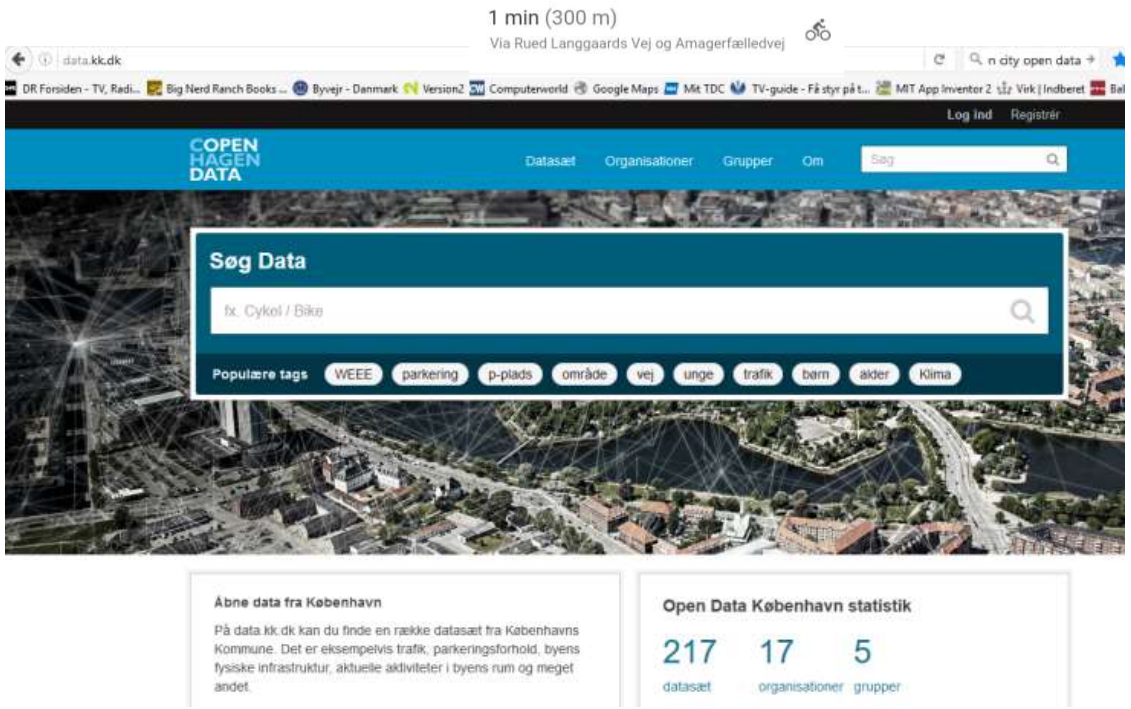
NetworkFetcher/JSON Parser

Current location

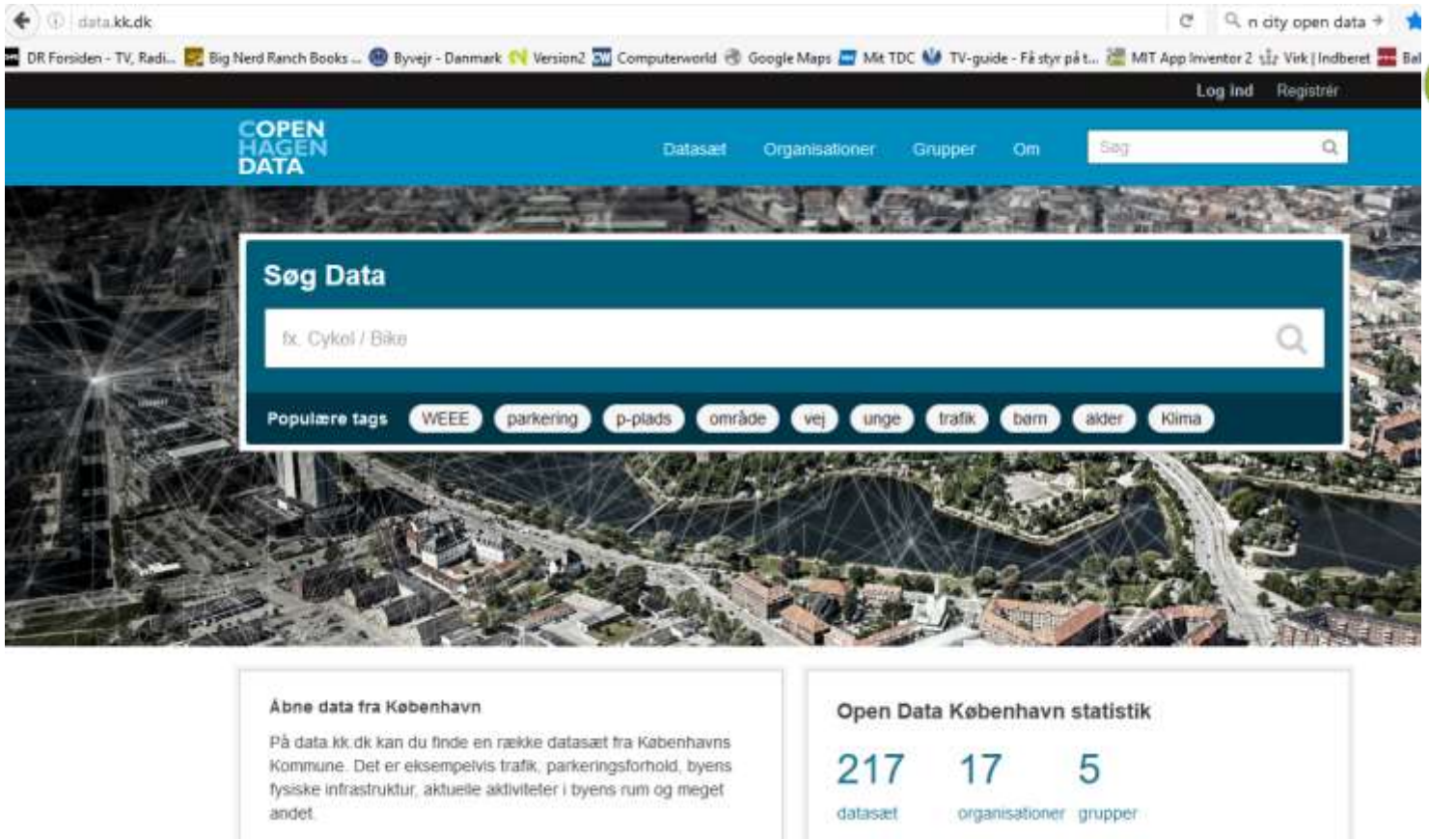
WebViewer

BikeStands Model





Copenhagen City Open Data



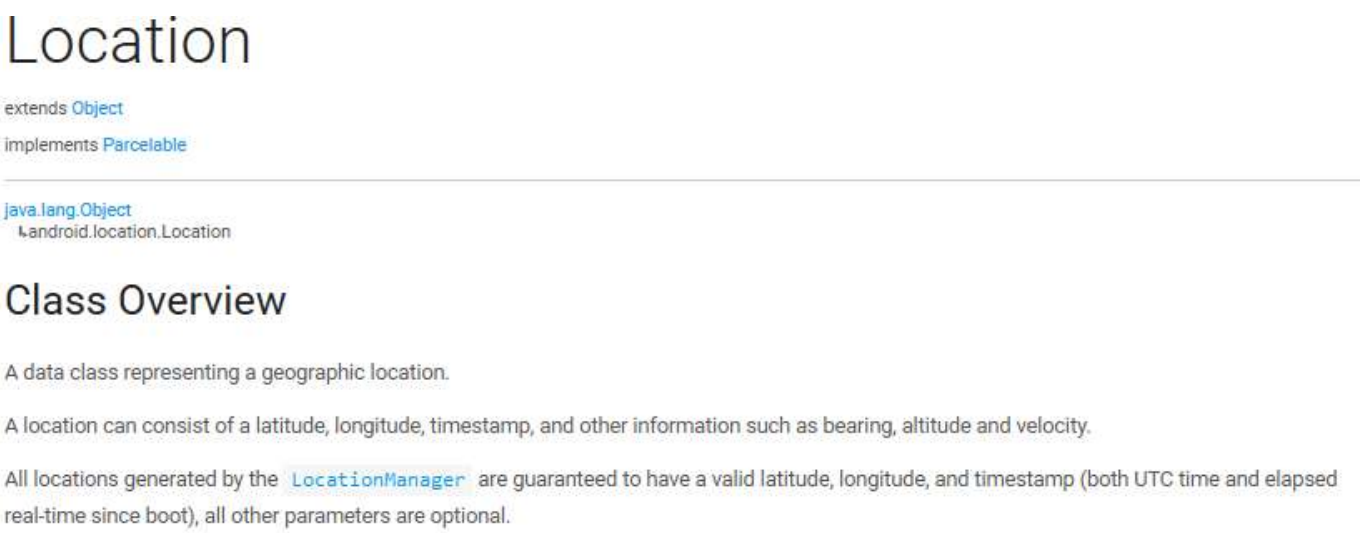
http://pro.jsonlint.com/

NetworkFetcher/JSONParser

```
public class NetworkFetcher {
 public byte[] getUrlBytes(String urlSpec) throws IOException { }
 public String getUrlString(String urlSpec) throws IOException {
        return new String(getUrlBytes(urlSpec));
 }
 public BikeStand[] fetchItems(String param) {
  try {
   String url = Uri.parse(param).buildUpon().build().toString();
   String jsonString = getUrlString(url);
   JSONObject jsonBody = new JSONObject(jsonString);
   return parseItems(jsonBody);
  } catch (JSONException je) {    …    }
 }
 private BikeStand[] parseItems(JSONObject jsonBody) throws … {
   JSONArray featureArray= jsonBody.getJSONArray("features");
   …
 }
}
```

BikeStand

```
public class BikeStand {
    private Location mLoc;  // GPS position
    private String mStreet; // Street name
    private int mPplaces;   // No of bike stands

    public BikeStand(Location loc, String street, int places) {
        mLoc= loc; …
    }
    …
}
```



Location

extends Object

implements Parcelable

java.lang.Object
↳android.location.Location

Class Overview

A data class representing a geographic location.

A location can consist of a latitude, longitude, timestamp, and other information such as bearing, altitude and velocity.

All locations generated by the `LocationManager` are guaranteed to have a valid latitude, longitude, and timestamp (both UTC time and elapsed real-time since boot), all other parameters are optional.

http://developer.android.com/reference/android/location/Location.html

Webviewer

http://maps.google.com?saddr=55.8225165,12.4228424&
daddr=55.72234925766947,12.482133029189042&dirflg=b

```
String url= "http://maps.google.com?saddr=" +
     start.getLatitude()+","+start.getLongitude()+
     "&daddr="+dest.getLatitude()+","+dest.getLongitude()+ "&dirflg=b”;
```

Use an implicit intent

```
private void startBrowser(Location start, Location dest) {
    Uri UrlPar= Uri.parse(url);
    Intent baseIntent = new Intent(Intent.ACTION_VIEW, UrlPar);
    startActivity(baseIntent);
}
```

Use WebView (textbook chapter 28)

```
Private WebView mWeb;

mWeb.loadUrl(url);
```

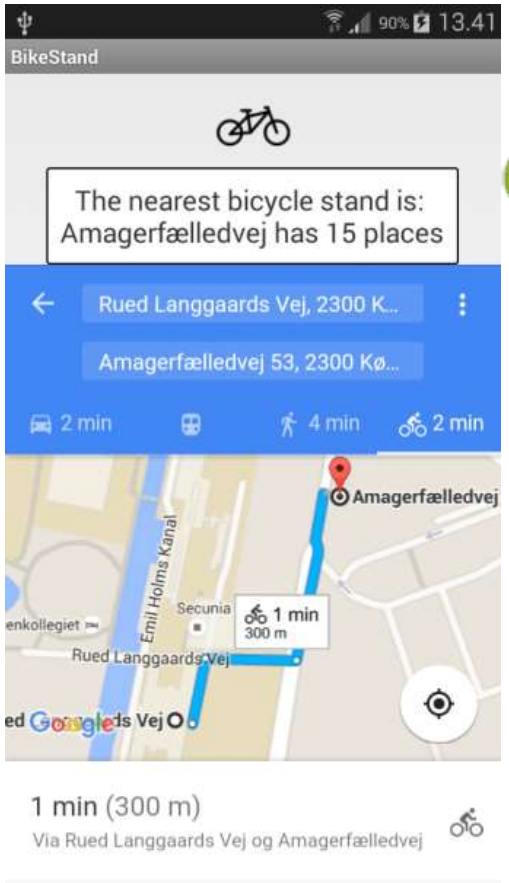
BikeStand app

NetworkFetcher/JSON Parser

Current location

WebViewer

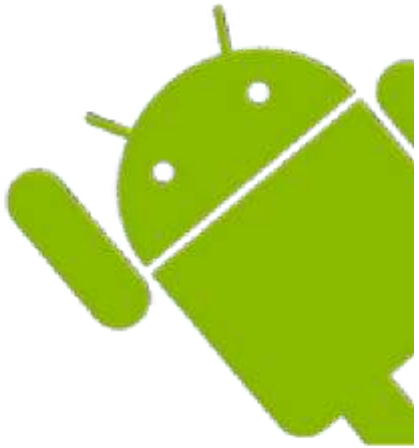
BikeStands Model



Calculating the distance between two locations p1 and p2?

```
p1.distanceTo(p2)
```

Google Play Services Location API



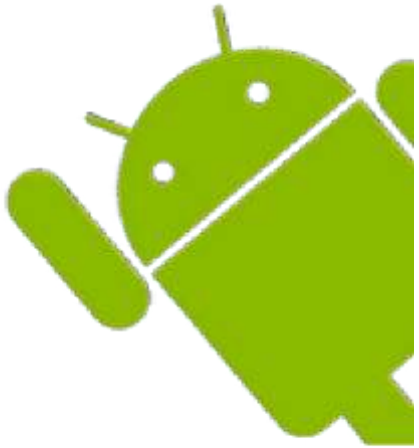
More sophisticated

One of many services available on the closed Google Play instead of open android.com

Requires registration and an API

https://developers.google.com/android/guides/setup#add_google_play_services_to_your_project

Android components



- **Activities**
- **Services**
- **Content Providers**
- **Broadcast Receivers**

Receive an **intent** that announces an event e.g.:

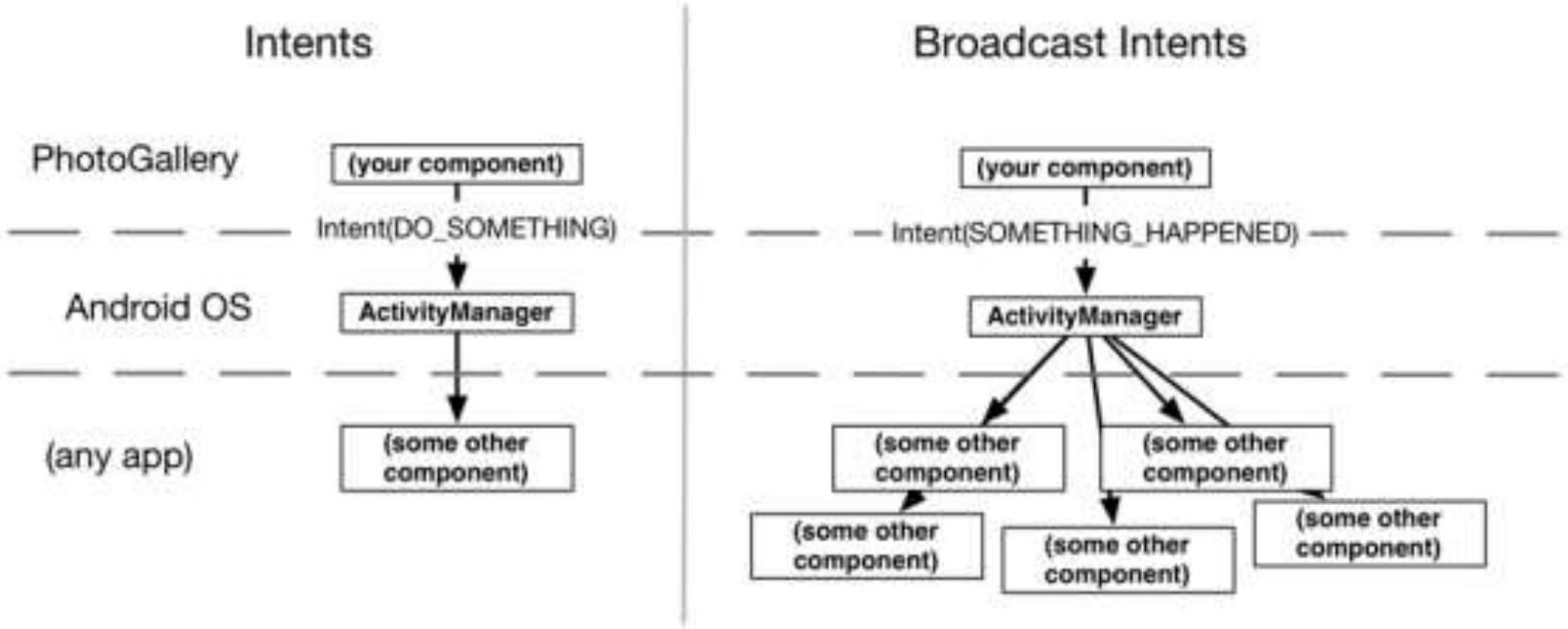
- the screen has turned off
- the battery is low, or
- a picture was captured
- wifi state changes
- SMS received

http://developer.android.com/reference/android/content/BroadcastReceiver.html

Broadcast intents

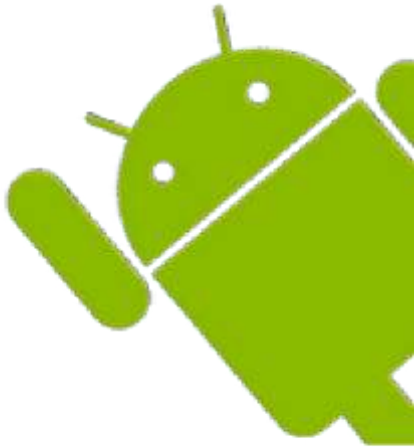
From textbook p. 491

Figure 27.1 Regular intents vs. broadcast intents



All receivers of the intent must specify this in their manifest

Two types of broadcasts



Normal: `sendBroadcast()`

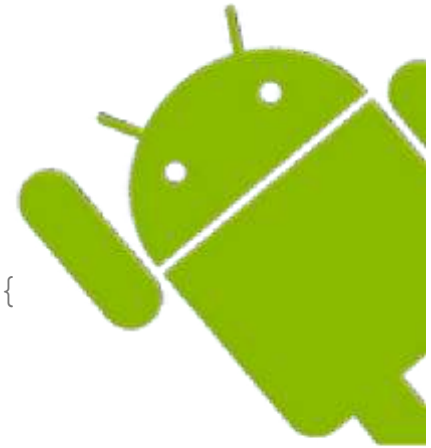
asynchrounous all receivers receive the broadcast and handle them in an undefined order

Ordered: `sendOrderedBroadcast()`

one receiver at a time. The order can be controlled with the ”android:priority” attribute of the matching intent filter.

```
<receiver android:name="com.pycitup.pyc.MyReceiver" >
    <intent-filter android:priority="1">
        <action android:name="com.pycitup.BroadcastReceiver" />
    </intent-filter>
</receiver>
<receiver android:name="com.pycitup.pyc.MySecondReceiver" >
    <intent-filter android:priority="2">
        <action android:name="com.pycitup.BroadcastReceiver" />
    </intent-filter>
</receiver>
```

Broadcast receiver example



Receiver class:

```
public class MyReceiver extends BroadcastReceiver {
  @Override
  public void onReceive(Context context, Intent intent) {
    Toast.makeText(context, "Received broadcast",
                            Toast.LENGTH_SHORT).show();
    }
}
```

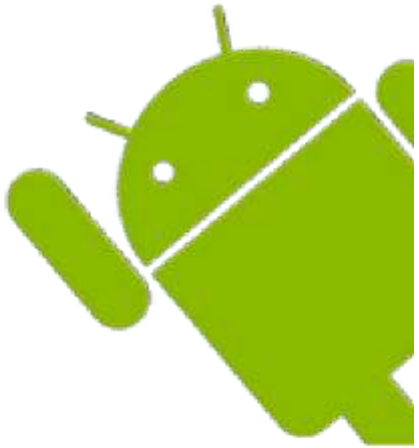
Static registration(Manifest):

```
<receiver
   android:name="dk.staunstrups.MyReceiver">
     <intent-filter>
         <action android:name="dk.staunstrups.MMAD" />
     </intent-filter>
</receiver>
```

Dynamic registration:

```
IntentFilter filter = new IntentFilter(”dk.staunstrups.MMAD");

MyReceiver myReceiver = new MyReceiver();
registerReceiver(myReceiver, filter);
```

Broadcast sender example

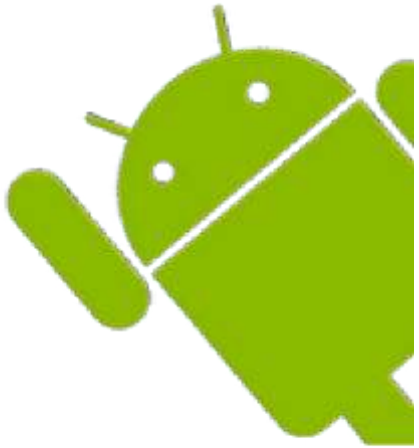


```
public class MainActivity extends Activity {

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



  public void doBroadcast(View view) {
    Intent intent = new Intent("dk.staunstrups.MMAD");
    sendBroadcast(intent);
  }
}
```

Example of system intents

Label	Description
android.intent.action.BATTERY_CHANGED	Sticky broadcast containing the charging state, level, and other information about the battery.
android.intent.action.BATTERY_LOW	Indicates low battery condition on the device.
android.intent.action.BATTERY_OKAY	Indicates the battery is now okay after being low.
android.intent.action.BOOT_COMPLETED	This is broadcast once, after the system has finished booting.
android.intent.action.BUG_REPORT	Show activity for reporting a bug.
android.intent.action.CALL	Perform a call to someone specified by the data.
android.intent.action.CALL_BUTTON	The user pressed the "call" button to go to the dialer or other appropriate UI for placing a call.
android.intent.action.DATE_CHANGED	The date has changed.
android.intent.action.REBOOT	Have the device reboot.

Course evaluation



Course is too simple/low level

Better coverage of databases and SQL

Lectures at 17:00

Lectures from Industry

Workload

Working on the same app throughout the course?

Second Mandatory Assignment

Feedback next week