Android Locations

Android Smartphone Programming

University of Freiburg

Matthias Keil / Tim Aicher
Institute for Computer Science
Faculty of Engineering
University of Freiburg

18. Dezember 2017

UNI
FREIBURG

- Internet connection useful for many applications.
- Examples: Loading ads or interact with *Google APIs* on a server.
- Application needs permission for internet usage[1].
- Extend application manifest:

```
1   <uses-permission android:name="android.
        permission.INTERNET" />
```
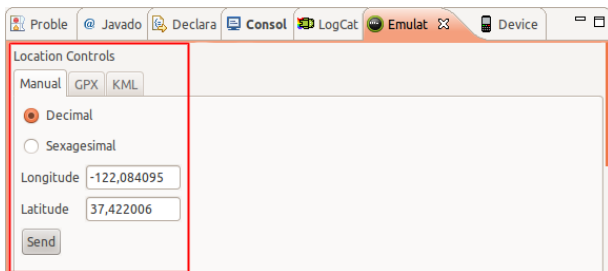
- GPS can be used to obtain user location.
- Problem: Works only outdoors and drains battery.
- Better: Android's *Network Location Provider*[4].
- Works indoors and outdoors, is faster and uses less battery power.
- Combination of both techniques also possible.
- Similar usage: Request location updates from *LocationManager* and what type of location provider to use.
  - *LocationManager.NETWORK_PROVIDER* for Network Location Provider.
  - *LocationManager.GPS_PROVIDER* for GPS Location Provider.

- Emulator in Android Studio can be fed mock GPS data.[5].
- Start emulator, then open emulator control.
- Can be found at *Tools* > *Android* > *Android Device Monitor* > *Emulator Control*.

UNI
FREIBURG

- *Geocoder* used to transform address into a coordinate (latitude and longitude)[2].
- *Reverse geocoding* is transforming a coordinate into an address.
- Needs internet permisson to access backend service.
- Part of the external *Google Maps library*.
- Can be given a locale to provide better results, for example *Locale.GERMANY* or *Locale.US*.
- Extend application manifest:

```
1   <uses-library android:name="com.google.
        android.maps" />
```

- Geocoding

```
1  List < Address > Geocoder . getFromLocationName (
       String locationName , int maxResults )
```

- Example string to describe location

```
1  " Georges - Koehler - Allee⎵101 ,⎵79110⎵Freiburg "
```

- Reverse geocoding

```
1  List < Address > Geocoder . getFromLocation ( double
       latitude , double longitude , int
       maxResults )
```

- Class *MapView* included in external Google Maps library[3].
- Easy way to display a map obtained from the Google Maps service.
- Enables similar user interaction as known from *http://maps.google.com*.
- *Maps API Key* needed to display data in a MapView object.
- Registration at Google Maps service mandatory to obtain the key.

- Example code can be found at *<sdk>/add-ons/google_apis-<api-level>/samples/MapsDemo*.
- Displaying a map using XML layout:

```
1  <LinearLayout xmlns:android="http://schemas.
      android.com/apk/res/android" ...>
2  <com.google.android.maps.MapView
3  android:id="@+id/map_view"
4  android:layout_width="..."
5  android:layout_height="..."
6  android:enabled="true"
7  android:clickable="true"
8  android:apiKey="mymapapikey" />
9  </LinearLayout>
```

- Obtaining coordinates from MapView: Override method
  dispatchTouchEvent

```
1   @Override
2   public boolean dispatchTouchEvent(MotionEvent
        ev){
3     if (ev.getAction() == MotionEvent.ACTION_UP
          ){
4       Projection p = mapView.getProjection();
5       GeoPoint loc = p.fromPixels((int)ev.getX
            (), (int)ev.getY());
6       Double longitude = ((double)loc.
            getLongitudeE6())/1000000;
7       Double latitude = ((double)loc.
            getLatitudeE6())/1000000;
8     }
9     return super.dispatchTouchEvent(ev);
10  }
```

- Class *Location* has a method *distanceTo(Location destination)* to approximate a distance in meters.

- Internet connection opens new possibilities for applications.
- User location can be obtained using a *Network Location Provider* and/or GPS.
- Mock GPS data can be created to test location functionality in emulator.
- A *Geocoder* translates between addresses and coordinates using an internet connection.
- External *Google Maps library* provides easy to use map functionality.

Bibliography
University of Freiburg

📄 ANDROID DEVELOPERS.
Connecting to the Network.
http://developer.android.com/training/basics/network-ops/connecting.html.

📄 ANDROID DEVELOPERS.
Geocoder Class Overview.
http://developer.android.com/reference/android/location/Geocoder.html.

📄 ANDROID DEVELOPERS.
Location and Maps.
http://developer.android.com/guide/topics/location/index.html.

📄 ANDROID DEVELOPERS.
Obtaining User Location.
http://developer.android.com/guide/topics/location/obtaining-user-location.html.

📄 ANDROID DEVELOPERS.
Providing Mock Location Data.
http://developer.android.com/guide/topics/location/obtaining-user-location.html#MockData.